PROBABILISTIC METHODS FOR DECISION MAKING IN PRECISION AIRDROP

A Dissertation
Presented to
The Academic Faculty

By

Andrew Leonard

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Mechanical Engineering

Georgia Institute of Technology

May 2019

PROBABILISTIC METHODS FOR DECISION MAKING IN PRECISION AIRDROP

Approved by:

Dr. Jonathan Rogers, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Adam Gerlach
Aerospace Systems Directorate
US Air Force Research Lab

Dr. Anirban Mazumdar
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Jun Ueda
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Fumin Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: January 9, 2019

To my family and friends.

ACKNOWLEDGEMENTS

I would first like to thank my advisor, Dr. Jon Rogers, for his time, effort, and mentorship over the past four and a half years. His commitment to research and thoughtful guidance to his students has made my time at Georgia Tech incredibly rewarding. From the research related discussions to visits at Wright-Patterson, I cannot think of a dull moment. A tremendous amount of thanks is also due to Dr. Adam Gerlach from the Air Force Research Lab. It has been a joy to work with you and I truly appreciate our discussions and your continued help and friendship over the past four years. I am sincerely grateful that I was able to work so closely with not only one, but two great people to produce this dissertation. I would also like to thank my committee for their time and insightful contributions to this work.

Thanks is also due to the many great friends I have made throughout my time at Georgia Tech. While our the grad student experience is what brought us together I like to think it is much more that kept us going. Specifically, I would like to thank the past and present members of the iREAL lab, including Dr. Jonathan Warner, Brian, Evan, Jared, Joey, and Kevin, for all of their help, friendship, and coffee runs over the years. Without you guys, and some other great friends, we could not have won the intramural ultimate frisbee championship. I cannot think of a better way to have ended my time at Georgia Tech than that.

Lastly, I would like to thank my closest friends and family, both in and out of Atlanta. The collective encouragement from you all has helped me push through some difficult times associated with this degree and I will never be able to thank you enough. Finally, I would like to thank my mom and dad for all of the love and support they have given me since the very beginning. From the phone calls, visits (including abroad), and even the occasional package of baked goods, I could not have arrived at this point without you two.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

It is inconvenient and often difficult to consider systems, or environments, that are not completely known. However, with the proliferation of autonomous systems, where the success or survival of the system is coupled to the ability to make decisions with incomplete information, it is becoming increasingly important. When restricting the scope of the system and its uncertainty to be linear and Gaussian, an efficient, closed-form solution can be constructed. However, for many complex systems, the reduction of the fidelity of both the system and the means to quantify its uncertainty may result in sub-par, or unintended, outcomes. With the advent of high-powered computing platforms, including the use of Graphics Processing Units (GPUs), to handle the increased computational workload, more generalized probabilistic planning methodologies may be explored.

This work introduces an optimal control algorithm in which the Koopman operator is used to solve for the probabilistically optimal input in the presence of initial condition and/or parametric uncertainty. The proposed approach is minimally restrictive; applicable to nonlinear systems and non-Gaussian uncertainty. Additionally, it offers unique computational advantages over alternative uncertainty quantification techniques, such as Monte Carlo methods, providing a practical method to compute a probabilistically optimal input. Leveraging kernel-based interpolation methods, the expected value computation may be carried out on large, high-dimensional scattered datasets.

In the context of the airdrop problem, these inputs are the optimal package release point, aircraft run-in, and transition altitude. Given an objective–or cost–function defined over the drop zone and a joint probability density function (PDF) quantifying the uncertainty in the system parameters and initial state, the objective function is pulled back to the drop altitude using the Koopman operator, and an expected value is computed with the joint PDF. The objective function provides a mechanism for tactical considerations to be included in the planning process. With the control inputs embedded in the initial joint PDF and the system

dynamics, the problem is treated as probabilistic from the beginning, a fundamental shift from the current state-of-the-art.

Simulation examples are presented highlighting performance of the algorithm in real-world scenarios. Included are a complex workload cost function over non-flat terrain and a scenario where a constraint must be satisfied probabilistically. Results compare favorably with those achieved through deterministic methods.

CHAPTER 1
INTRODUCTION

1.1 Problem Statement & State-of-the-Art

As autonomous systems evolve toward increasing use in practical environments, they naturally encounter growing uncertainty in dynamic behavior and the environment in which they operate. There is a growing consensus that such systems must be able to derive optimal control inputs based not on the mean behavior of the system, but rather on a probabilistic model of the system incorporating all aspects of uncertainty. A variety of stochastic optimal control algorithms have been developed to address this growing need. Many stochastic optimal control algorithms seek to maximize (minimize) an expected value between the time-varying joint PDF of the system state and an objective (cost) function. The control inputs that minimize this cost represent the optimal inputs given the assumed uncertainty distributions.

There has been extensive theoretical developments in the field of stochastic optimal control over the previous several decades. Given a mathematical description of the system, the methodology to optimize a single, or possibly multiple, control decisions probabilistically against an objective, or cost, function is sought. A well-known result for linear systems with additive Gaussian noise against a quadratic cost function is the Linear Quadratic Gaussian (LQG) control law [1], which minimizes the expected value of the cost over an infinite or finite horizon [2]. The LQG solution represents the pairing of the Kalman Filter [3, 4], a linear-quadratic estimator (LQE), and the linear-quadratic regulator (LQR) [5, 6]. For nonlinear systems and/or non-Gaussian uncertainty, the time-varying density oftentimes cannot be evolved in closed-form. Numerous control algorithms have been proposed to date for these types of problems to include path integral control [7], dynamic program-

ming [8], Monte Carlo-based methods [9, 10], and others [11, 12].

A key element of many nonlinear stochastic optimal control algorithms is an uncertainty propagation component, which estimates the time-evolution of the joint PDF of the system. A variety of methods exist for this purpose including Monte Carlo, Polynomial Chaos [13], direct quadrature method of moments [14], direct evolution through the Fokker-Planck-Kolmogorov equation [15], and, for linear Gaussian systems, direct evolution of the covariance through the Ricatti equation [16]. Comparing each method, there exists a trade-off between the generality of the problem that it may be applied to and the ability to obtain a solution, whether it be in closed-form or evaluated numerically. Considering aspects of the problem such as nonlinearity, non-Gaussian uncertainty, process noise, and dimensionality, the number of usable methods begin to diminish.

Airdrop is one such application in which decisions must be made in the presence of significant uncertainty in system behavior and the environment. Over the past several decades, shifts in military tactics and logistical environments have led to increasing interest in the use of airdrop missions for military resupply and large-scale humanitarian aid. A variety of technology development efforts have been pursued which have focused on improving accuracy, practicality, and safety of airdrop systems. Generally, two types of airdrop systems are currently in use: guided systems, in which the package is actively controlled under a rectangular parafoil; and unguided systems, which typically fall ballistically under a circular parachute. Guided systems exhibit superior accuracy, but the cost associated with such systems (and the need to recover their guidance units) precludes their use in large-scale delivery missions and in contested areas [17]. For unguided systems, accuracy considerations usually dictate that packages should be released from a low altitude to limit the packages' exposure to uncertain winds. However, in response to the proliferation of threats from small arms fire, anti-aircraft artillery, and Man Portable Air Defense Systems, precision airdrop missions are shifting to High Altitude, Low Opening (HALO) procedures to better protect aircraft and flight personnel [18]. In HALO drops, packages are released from a

high altitude under a drogue parachute, falling at a high descent rate. The main parachute is deployed at a lower altitude to slow the system before impact with the ground.

Given the absence of feedback control on ballistic airdrop systems, there is limited opportunity to improve accuracy. One avenue that has been considered is to optimize the transition altitude at which the main parachute is inflated [19, 20]. This affords limited control authority with which to reduce dispersion, allowing packages to optimize their impact location along a so-called "line of control" [20] determined by the wind profile. Another opportunity for accuracy improvement in ballistic airdrop lies in improved selection of the Computed Air Release Point (CARP), the point where the packages are released from the aircraft, and from a secondary standpoint, optimization of the aircraft heading on arrival at the CARP. The US military's current airdrop mission planning tool, the Consolidated Airdrop Tool (CAT) [21], uses a nominal model of the wind field (based on a current wind forecast), a nominal package descent rate, and an intended point of impact (IPI) to solve for the optimal CARP location through a boundary value solution process. Monte Carlo simulations are then performed from this CARP to estimate the resulting dispersion caused by uncertainty in winds, package descent rates, aircraft roll-out variation, and other factors. From the resulting dispersion pattern the user must make a go/no-go decision. If the outcome is unacceptable, the planning process must be restarted with a new IPI. Given this deterministic solution method for the CARP, limited accuracy enhancements are possible by improving the planner's internal models of package release [22] and parachute inflation [23, 24] as well as the wind forecast accuracy [25, 26].

1.2 Objective of Work

The focus of this work is to develop a methodology applicable to a class of problems that require the probabilistic optimization of control decisions under system behavior and environmental uncertainty. As such, the optimization problem is framed using an expected value to minimize a cost function, or maximize an objective function. The algorithms pre-

sented in this paper leverage explicit uncertainty propagation techniques [27, 28, 9, 29] to compute the optimal control decisions in the presence of initial condition uncertainty. This explicit propagation can be performed through density evolution techniques such as the Liouville Equation, a first-order ODE derived from the Frobenius-Perron operator, as described in [20]. However, there are unique computational advantages obtained by using the adjoint to this forward propagation scheme, i.e. the Koopman operator. Here, the Koopman operator is instead used to *pull back* the objective function to the initial time, at which the expected value can be maximized (minimized) with respect to the initial joint density. While the Frobenius-Perron and Koopman operators are well-known in mathematics-focused dynamical systems literature [30, 31], it has not been until recently that they have been applied to more tangible engineering problems, e.g. [32, 33, 34]. However, while these works exploit certain properties of the Frobenius-Perron and Koopman operators individually, they do not explicitly utilize the adjoint relationship between the two. As such, it a goal of this thesis to present the operators, with an emphasis on their adjoint relationship, for their use in the planning framework such that the application-oriented reader may find them beneficial. Leveraging the adjoint property of the operators has been explored in the context control optimization and is therefore a novel contribution of this work.

The mission planning methodology presented in this paper represents a fundamental paradigm shift in how airdrop planning is to be performed, directed toward approaches that explicitly account for uncertainty and away from approaches that use only nominal (or mean) information about the system in the planning process. The advantage of this new probabilistic planning approach is that the uncertainty in all aspects of the airdrop process are considered as inputs to the CARP computation process and are directly used in its selection, rather than used strictly to evaluate mission feasibility. Along the lines of the work presented in [20], in this new probabilistic framework the user provides an objective function quantifying mission success as a function of drop zone impact location. This definition of mission success can be much more general than a classical definition

of a single IPI, and can represent operational factors such as retrieval workload [35]. As a result, the CARP location is selected as that which optimizes the multi-variate criteria for mission success given the uncertainty distributions for a given scenario. This ability to rapidly generate control solutions according to multi-variate success criteria is a key element of Operational Agility as outlined in the Air Force Future Operating Concept [36].

Applying the control optimization methodology to the precision airdrop problem, an objective function is defined over the drop zone which encodes information about the desired impact point, obstacles, and other tactical considerations. Uncertainty distributions are defined capturing uncertainty in winds, parachute dynamics, and aircraft release conditions, which are partially parameterized by the selected CARP and run-in. The goal of the mission planner is then to optimize the CARP, run-in heading, and, possibly, transtion altitude such that the expected value between the objective function and the joint uncertainty distribution propagated to ground impact is maximized (or minimized). This solution for CARP and run-in represents the probabilistically optimal release location for the packages which will generate the highest (or lowest) average value of the objective function, given the assumed uncertainty. When the problem considers the transition altitude as a variable to be optimized, the solution also includes a transition altitude schedule that returns the altitude at which the main parachute should be inflated to obtain the optimal objective or cost value. Note that in this framework the planning and uncertainty validation steps, previously separated into two different stages by the CAT planning tool, are now combined into a single step in order to generate a probabilistically optimal solution.

It is important to understand that the probabilistic optimization methodology described in this paper may be used across a wide variety of engineering problems and is not limited to the airdrop application. Recently, numerous authors have investigated the use of explicit uncertainty quantification techniques, specifically the Liouville Equation (which is the transport equation of the Frobenius-Perron operator), as an alternative to Monte Carlo simulation to evolve a probability density through time for the purposes of trajec-

tory optimization [37], system performance analysis [38], and other applications [39, 40]. Several computational issues often arise in implementation of the Liouville Equation, however. These include problems with numerical precision caused by the rapid reduction in the PDF support for many systems [41] and the need for scattered data integration over a non-uniform mesh of points at which the PDF is known. For problems in which the expectation of the time-varying PDF must be computed, the algorithm presented here leverages the Koopman operator to circumvent these numerical issues. By ”pulling back” the objective function at the initial time through the Koopman operator and evaluating the expectation over the initial joint density, the underlying mesh on which numerical integration must be performed can be designed advantageously to reduce numerical inaccuracies. This solidifies the novel use of the Frobenius-Perron-Koopman adjoint relationship to solve the problem of control selection under uncertainty.

1.3 Structure of Work

The structure of this work proceeds as follows. First, relevant background on uncertainty quantification and density evolution through a dynamical system is presented. This includes a discussion of the Frobenius-Perron and Koopman operators. The discussion highlights assumptions and restrictions of the operators and focuses primarily on their application to systems of ordinary differential equations. In the same chapter, kernel-based methods for scattered-data interpolation and integration are presented. The use of Lobachevsky splines for use in computing conditional expectations, or marginalizations, is investigated and included in an example. Next, the Koopman operator approach to optimization in the presence of uncertainty is presented, to include a discussion of the numerical advantages in implementation compared to the Frobenius-Perron operator. The methodology is developed in a general sense, considering an arbitrary system and associated control decisions.

Following the generalized optimization methodology, the focus is shifted to its application to the unguided precision airdrop problem. As such, the dynamic model and

corresponding sources of uncertainty in the drop and descent procedures are identified. The proposed mission planning algorithm is presented in terms of each of the relevant computational steps. This includes frameworks for CARP and run-in optimizations for a predetermined and optimized transition altitude. Additionally, an algorithm for real-time transition altitude optimization during descent is developed. Example simulation results are presented to demonstrate the capabilities of the developed methods in complex drop environments, including performance comparisons against current deterministic methods. Finally, implications of this work and avenues for additional developments are discussed in the Conclusion.

CHAPTER 2
MATHEMATICAL TOOLS

2.1 Frobenius-Perron & Koopman Operators

Given that, in general, some aspects of a dynamical system are not completely known, it is beneficial to understand how variations in the initial state or system parameters affects the state time histories of the system. When this variation is captured by a density and the dynamical system is described by a set of ordinary differential equations (ODEs), this propagation can be performed through the use of well-known uncertainty quantification methods based on the Frobenius-Perron operator. However, for certain classes of problems the adjoint to the Frobenius-Perron operator, the Koopman operator, proves to be advantageous. For a thorough development of these tools the interested reader should consult [30, 32, 42], and [43] for an ODE-specific treatment.

2.1.1 Frobenius-Perron Operator

The Frobenius-Perron operator provides a method to propagate initial condition uncertainty through a system. More generally, it is a mechanism that allows for the measure-preserving transformation of a density through a map. As such, it is applicable to both continuous and discrete systems that satisfy certain conditions that will be elaborated on below. The focus of this section is to layout the formulation of the operator, highlighting important characteristics. A rigorous derivation including proofs may be found in [30].

To begin, a measure space $(\Omega, \mathcal{A}, \mu)$ is prescribed over the state-space $\mathbb{R}^d$. For the purposes of this work $\mathcal{A}$ is taken as the Borel σ-algebra of $\mathbb{R}^d$ and μ the Borel measure [44]. If a mapping $\mathrm{S}\colon \Omega \to \Omega$ is applied over the state-space, elements $\boldsymbol{x} \in \Omega$ are transformed according to $\boldsymbol{x}_2 = \mathrm{S}(\boldsymbol{x}_1)$. The same holds true for densities over the state-space, however

this transformation is handled by the Frobenius-Perron operator of S, $\mathrm{P_S}$. Written in a similar fashion, the densities f are transformed according to

$$f_2 = \mathrm{P_S} f_1 \qquad f_1,\, f_2 \in L^1 \tag{2.1}$$

where $\mathrm{P_S}\colon L^1 \to L^1$. The densities, or functions, f_1 and f_2 are members of L^1, the space of Lebesque integrable functions satisfying $\int_\Omega \left| f\left(\boldsymbol{x}\right)\right| \mu\left(d\boldsymbol{x}\right) < \infty$. Additionally, if $f\colon \Omega \to \mathbb{R}$ is considered to be non-negative for all $\boldsymbol{x} \in \mathbb{R}^d$ and is restricted such that $\int_\Omega f\left(\boldsymbol{x}\right) \mu\left(d\boldsymbol{x}\right) = 1$, it is then a probability density, or probability density function (PDF) [45].

Given a density f and known measure μ, a new measure on $A \in \mathcal{A}$ may be *induced* by f according to

$$\mu_f(A) = \int_A f(\boldsymbol{x}) \mu(d\boldsymbol{x}) \tag{2.2}$$

If f is a PDF, then Eq. (2.2) is the process of assigning probabilities to subsets of the state-space and the induced measure is a probability measure. The process may also be reversed using the Radon-Nikodym theorem [46], which states that if a second measure v is known and $v(A) = 0$ for all $A \in \mathcal{A}$ where $\mu(A) = 0$, then the density f is uniquely defined by

$$v(A) = \int_A f(\boldsymbol{x}) \mu(d\boldsymbol{x}) \tag{2.3}$$

Leveraging Eqs. (2.2) and (2.3), the Frobenius-Perron operator of the mapping S, denoted $\mathrm{P_S}$, is defined as the operator that satisfies the equality

$$\int_A \mathrm{P_S} f(\boldsymbol{x}) \mu(d\boldsymbol{x}) = \int_{\mathrm{S}^{-1}(A)} f(\boldsymbol{x}) \mu(d\boldsymbol{x}) \quad \forall\, A \in \mathcal{A} \tag{2.4}$$

where $\mathrm{S}^{-1}(A)$ is the counter-image of A. The mapping S must be both measurable, $\mathrm{S}^{-1}(A) \in \mathcal{A}\ \ \forall\, A \in \mathcal{A}$, and nonsingular, $\mu\left(\mathrm{S}^{-1}(A)\right) = 0\ \ \forall\, A$ such that $\mu(A) = 0$ [30]. As such, the operator is uniquely defined by Eq. (2.4) and conserves the measure of element $\mathrm{S}^{-1}(A)$–

the right-hand side of Eq. (2.4)–as it is mapped to A by S. This is accomplished by the right-hand side integral of Eq. (2.4) being simultaneously the induced measure of Eq. (2.2) *and* the known measure in Eq. (2.3). Substituting in Eq. (2.1) for the density in Eq. (2.3), and connecting the integrands by the mapping S, completes the equality.

From a more intuitive viewpoint, considering that f is a PDF, the Frobenius-Perron operator conserves the *probability mass* of the subregion A when it is transported to a new region of the state space according to S. This is true for any possible subregion in the domain Ω. When $A = \Omega$, the total probability mass is conserved. If the mapping S is differentiable and invertible, then Eq. (2.4) may be differentiated to explicitly solve for the transformed density function

$$\mathrm{P}_{\mathrm{S}} f(\boldsymbol{x}) = f\left(\mathrm{S}^{-1}(\boldsymbol{x})\right) \left| \frac{d\mathrm{S}^{-1}(\boldsymbol{x})}{d\boldsymbol{x}} \right| \tag{2.5}$$

where $\left| d\mathrm{S}^{-1}(\boldsymbol{x}) \,/\, d\boldsymbol{x} \right|$ is the determinant of the Jacobian of the inverse mapping S^{-1}.

In many situations, it is of interest to apply the mapping S iteratively, say in the case of a discrete-time system. If S^n is the mapping S applied to the state-space n times, $\mathrm{S}^n = \mathrm{S} \circ \cdots \circ \mathrm{S}$, then, using its linearity, the Frobenius-Perron operator to transform f by the mapping S^n is $(\mathrm{P}_{\mathrm{S}})^n$. When the separation (time) between consecutive applications approaches zero, the iterated Frobenius-Perron operator may be expressed using its *infinitesimal* form. This is of benefit when system dynamics are described by set of ODEs, as is the case when considering autonomous, continuous-time systems of the form

$$\frac{d\boldsymbol{x}}{dt} = F(\boldsymbol{x}) \tag{2.6}$$

where $F \in C^1$ to guarantee the existence and uniqueness of solutions for all $t \geq 0$. As such, a mapping parameterized by time may be written as

$$\boldsymbol{x}(t) = \mathrm{S}_t(\boldsymbol{x}_0) = \boldsymbol{x}_0 + \int_0^t F\left(\boldsymbol{x}(\tau)\right)\, d\tau \tag{2.7}$$

where the time-invariance of F allows for the property that $\mathrm{S}_t(\mathrm{S}_{t'}(\boldsymbol{x})) = \mathrm{S}_{t+t'}(\boldsymbol{x})$. Considering also that $\mathrm{S}_0(\boldsymbol{x}) = \boldsymbol{x}$ and F is continuous, the set of all possible mappings $\{\mathrm{S}_t\}_{t\geq 0}$ generated from the set of ODEs in Eq. (2.6) is known as a semidynamical system, with corresponding semigroup of Frobenius-Perron operators $\{\mathrm{P}_t\}_{t\geq 0}$ [30].

Semigroups generated by systems of ODEs may be equipped with an infinitesimal operator. The Frobenius-Perron infinitesimal operator of $\{\mathrm{P}_t\}_{t\geq 0}$ on the density f is defined in the limit as

$$A_{FP}f = \lim_{t\to 0} \frac{P_t f - f}{t} \tag{2.8}$$

and in [30], with help from the Koopman operator, is found to be equal to

$$A_{FP}f(\boldsymbol{x}) = -\sum_{i=1}^{d} \frac{\partial(f\, F_i)}{\partial x_i}(\boldsymbol{x}) \tag{2.9}$$

where d is the dimension of the state space. For the density f, the function $u(t,\, \boldsymbol{x}) = \mathrm{P}_t f(\boldsymbol{x})$ satisfies $u'(t,\, \boldsymbol{x}) = A_{FP}u(t,\, \boldsymbol{x})$ and is a solution to the partial differential equation

$$\frac{\partial u}{\partial t} + \sum_{i=1}^{d} \frac{\partial(uF_i)}{\partial x_i} = 0 \tag{2.10}$$

with the initial condition $u(0,\, \boldsymbol{x}) = f(\boldsymbol{x})$.

The partial differential equation (PDE) of Eq. (2.10) may be solved using the Method of Characteristics (MOC) [47]. By application of the MOC, the PDE in Eq. (2.10) collapses to a first-oder ODE that is valid only along trajectories of the system in Eq. (2.6), i.e,

$$\frac{du}{dt} = -u(t)\Phi(\boldsymbol{x})\Big|_{\boldsymbol{x}(t)} \tag{2.11}$$

where $\Phi(\boldsymbol{x}) = \sum_{i=1}^{d} \partial F_i(\boldsymbol{x}) \,/\, \partial x_i$ is the trace of the Jacobian of the dynamics $F(\boldsymbol{x})$. Equation (2.11), also known as the Stochastic Liouville Equation (SLE) [48], may be solved alongside the ODEs in Eq. (2.6) yielding a representation of the time-dependent density $u(t,\, \boldsymbol{x})$. As it relates to an initial value problem (IVP), the Frobenius-Perron operator is a

mechanism that allows for uncertainty–quantified by a PDF–to be *pushed forward* through a dynamical system with equations of motion (EOMs) defined by F.

2.1.2 Koopman Operator

The Koopman operator [31] is a mechanism that allows for the transportation of a function through a map, or dynamical system. Also known as the composition operator, the Koopman operator $\mathrm{U}\colon L^\infty \to L^\infty$ states that

$$\mathrm{U}_\mathrm{S} g(\boldsymbol{x}) = g\left(\mathrm{S}(\boldsymbol{x})\right) \quad \forall g \in L^\infty \tag{2.12}$$

where S is non-singular and L^∞ is the space of functions satisfying $\|g\|_\infty < \infty$. Intuitively, g, a function of the state-space vector $\boldsymbol{x}$, may be considered as an *observable* of the state-space.

Considering again a system of ODEs generating a set of state-space mappings, Eq. (2.7), the semidynamical system $\{\mathrm{S}_t\}_{t\geq 0}$ also has a corresponding semigroup of Koopman operators, $\{\mathrm{U}_t\}_{t\geq 0}$. Like the Frobenius-Perron operator, the semigroup of Koopman operators may be equipped with an infinitesimal form. Assuming a continuously differentiable g with compact support, the infinitesimal Koopman operator A_K is explicitly derived from the limit [30]:

$$A_K g = \lim_{t\to 0} \frac{\mathrm{U}_t g(\boldsymbol{x}_0) - g(\boldsymbol{x}_0)}{t} = \sum_{i=1}^{d} \frac{\partial g(\boldsymbol{x}_0)}{\partial \boldsymbol{x}_i} F_i(\boldsymbol{x}_0) \tag{2.13}$$

and is valid for all $\boldsymbol{x}_0$. Following the form of the infinitesimal Frobenius-Perron operator's derivation, the function $u(t,\, \boldsymbol{x}) = \mathrm{U}_t\, g(\boldsymbol{x})$ is taken to satisfy $u'(t,\, \boldsymbol{x}) = A_K u(t,\, \boldsymbol{x})$ and is a solution to the partial differential equation

$$\frac{\partial u}{\partial t} - \sum_{i=1}^{d} F_i \frac{\partial u}{\partial x_i} = 0 \tag{2.14}$$

with the initial condition $u(0,\, \boldsymbol{x}) = g(\boldsymbol{x})$.

Using the Method of Characteristics once again, Eq. (2.14) collapses to a first-order ODE that is valid only along trajectories of the system:

$$\frac{du}{dt} = 0 \Big|_{\boldsymbol{x}(t)} \tag{2.15}$$

This may be solved alongside the system ODEs in Eq. (2.6). Equation (2.15) states that, given a deterministic mapping, the initial conditions $\boldsymbol{x}_0$ completely define the value of $g(\boldsymbol{x}(t))$ along the trajectory from $\boldsymbol{x}(0) \to \boldsymbol{x}(t)$.

The necessity of a continuously differentiable and compactly supported (which is, informally, that outside of a closed domain the function is identically zero) g, and therefore its derivatives, asserts the convergence of the difference quotient as $t \to 0$ used in deriving Eq. (2.14) [30]. When considering functions that are not naturally compactly supported, by including a maximum bound, the function may be saturated and inverted to ensure compact support. Conversely, after applying the Koopman operator, the original function may be recovered within the domain of compact support. The inversion and reversion steps are defined as:

$$\text{Invert}: \quad g_{\text{c}}(\boldsymbol{x}) = \max\big(\Lambda - g(\boldsymbol{x}),\, 0\big) \qquad \forall \boldsymbol{x} \in \Omega \tag{2.16a}$$

$$\text{Revert}: \quad g(\boldsymbol{x}) = \Lambda - g_{\text{c}}(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in A_X \tag{2.16b}$$

where Λ is the maximum bound, g_c is compactly supported, and $A_X \subset \Omega$ is the support of g_{c}.

2.1.3 Adjoint Relationship

The Frobenius-Perron and Koopman operators of a mapping S, noted $\mathrm{P_S}$ and $\mathrm{U_S}$, respectively, are adjoint to each other and satisfy the equality [30]

$$\langle \mathrm{P_S} f,\, g \rangle = \langle f,\, \mathrm{U_S} g \rangle \qquad f \in L^1 \text{ and } g \in L^\infty \tag{2.17}$$

where f and g are continuously differentiable and g is compactly supported.

The adjoint relationship with the Frobenius-Perron operator provides a convenient and intuitive use of the Koopman operator. As the inner products in Eq. (2.17) are operating on real-valued functions defined on the state-space Ω, they can be rewritten in the following form [49]

$$\langle u, v \rangle = \int_\Omega u(\boldsymbol{x})\, v(\boldsymbol{x})\, d\boldsymbol{x} \tag{2.18}$$

If $\boldsymbol{x}$ in Eq. (2.18) is a random variable emitting a PDF defined by u, and v is a function of $\boldsymbol{x}$, then the integral on the right-hand side of Eq. (2.18) is equivalent to the expected value of $v(\boldsymbol{X})$ when $\boldsymbol{X} \sim u$, i.e.

$$\mathbb{E}\left[v(\boldsymbol{X})\right] = \int_\Omega u(\boldsymbol{x})\, v(\boldsymbol{x})\, d\boldsymbol{x} = \langle u, v \rangle \tag{2.19}$$

Figure 2.1 provides a visual representation of this adjoint relationship, leveraging the insight of Eq. (2.19). Consider a dynamical system with initial condition uncertainty. The top row of Figure 2.1 represents the left side of the adjoint equation, Eq. (2.17). The PDF f (dashed line) is propagated through the system dynamics and an inner product is taken with g (solid line). The scalar result–the expected value of g under the uncertainty $\mathrm{P}f$–is represented by the area of the shaded region. The bottom-row represents the right side of the adjoint equation, where the Koopman operator is applied to g before the inner product with f is taken. The scalar result, the shaded region, in this case is the expected value of $\mathrm{U}g$ under the uncertainty f. While the pointwise-products $\big(\mathrm{P}f(\boldsymbol{x}) \cdot g(\boldsymbol{x})\big)$ and

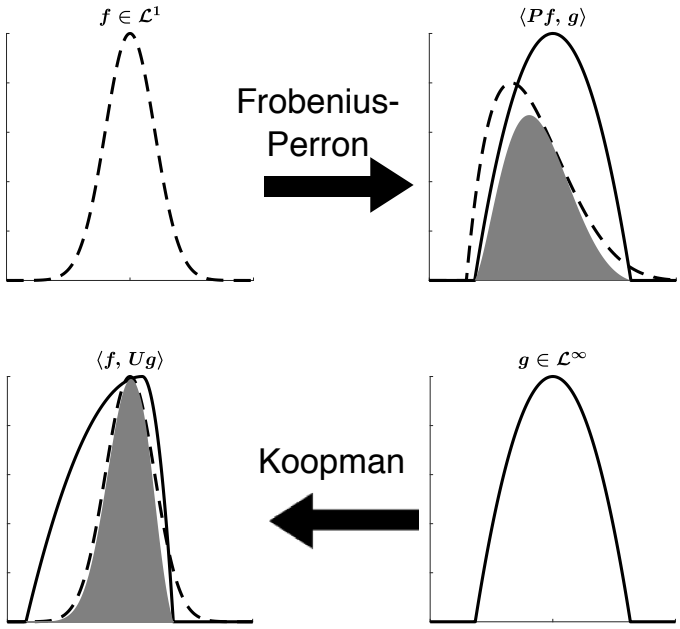


Figure 2.1: Illustration of the Frobenius-Perron and Koopman operators adjoint property. The inner products, represented by the area of the filled regions in the top-right and bottom-left plots, are equivalent

$\big(f(\boldsymbol{x}) \cdot \mathrm{U}g(\boldsymbol{x})\big)$ may not be equal everywhere-as seen by the differing shapes of the shaded regions–the scalar result of the inner products, i.e., the total shaded areas, are equal [30].

As a result, it is convenient to consider the Frobenius-Perron operator as a *push-forward* mechanism on measures and the Koopman operator as a *pull-back* mechanism on observations, both inherently dependent on the dynamics of the underlying system. Through their adjoint relationship, the equality of the inner product shown in Eq. (2.17), the operators may be used to express two different, but equivalent, forms of an expected value calculation.

2.1.4 Parameter Uncertainty & Process Noise

Interpreting the adjoint relationship of the Frobenius-Perron and Koopman operators as an expected value provides a means for it to be used in problems that contain uncertainty in initial conditions of systems. In many engineering problems, there is uncertainty in not only the initial conditions but also parameters defining the specific behavior of the system.

To allow for the Frobenius-Perron and Koopman operators to be applied to these systems, consider a dynamical system of the form found in Eq. (2.6),

$$\frac{d\boldsymbol{s}}{dt} = F\left(\boldsymbol{s} \mid \boldsymbol{p}\right) \tag{2.20}$$

$$F : \mathbb{R}^n \to \mathbb{R}^n \tag{2.21}$$

where $\boldsymbol{s} \in \mathbb{R}^n$ is the state vector, $\boldsymbol{p} \in \mathbb{R}^p$ is a vector containing possible uncertain parameters, and F is a continuous function such that solutions to Eq. (2.20) exist and are unique [30]. Alternatively, this system can be represented with the parameters included as additional states with no dynamics [42]:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{p} \end{bmatrix} \tag{2.22}$$

$$\frac{d\boldsymbol{x}}{dt} = \bar{F}(\boldsymbol{x}) = \begin{bmatrix} F\left(\boldsymbol{s} \mid \boldsymbol{p}\right) \\ \boldsymbol{0}^{p\times 1} \end{bmatrix} \tag{2.23}$$

where $\bar{F}\colon \mathbb{R}^{n+p} \to \mathbb{R}^{n+p}$. By using this augmented system, all parametric uncertainty can be treated as initial condition uncertainty, captured by a single joint PDF on the augmented state-space, $f(\boldsymbol{x})$. For systems with parametric uncertainty, the Frobenius-Perron and Koopman operators will be of the augmented system form in Eq. (2.23).

As stated in Section 2.1.1, the Frobenius-Perron operator conserves probability mass for every $A \in \mathcal{A}$, including total probability when $A = \Omega$. As such, the operator is applicable to initial condition uncertainty only and may not include process noise within the mapping (probability mass may not be created or added). However, Halder et. al [50] have shown that by representing the stochastic process as an infinite (or truncated) combination of deterministic functions using the Karhunen Loève expansion, process noise may be added to a Frobenius-Perron operator in the form of parametric uncertainty. However, in doing so, the overall dimension of the augmented system is increased.

2.2 Scattered Data Interpolation & Integration

This section presents the problem of, and possible solutions to, multivariate, scattered data integration via interpolation. When computing the infinitesimal forms of the Frobenius-Perron and Koopman operators along *trajectories* using the MOC, the resulting densities and functions, respectively, are typically in the form of scattered data and must be interpolated to produce a continuous representation. The notation of *scattered* data is that it has no prefixed structure e.g., gridded or sampled uniform, normally, etc. To begin the section, the problem is first defined and then a naive approach is given to reinforce the necessity of better solutions. Next, the idea of kernel-based methods is introduced. Radial Basis Functions (RBF), a well-known kernel-based method, are explored as a possible solution. Following the RBFs, Lobachevsky splines, another kernel-based method, are investigated. Finally, an example using software developed for–and fundamental to–this work is given. In line with the overall goal of this work, this section hopes to bridge the gap between the applied math literature and tangible engineering problems.

2.2.1 Problem Statement

The scattered data interpolation (approximation) problem can be stated as follows: given a set of data $\{(\boldsymbol{x}_i,\, y_i)\}_{i=1}^{N}$ where $\boldsymbol{x} \in \mathbb{R}^d$, $y \in \mathbb{R}$, and $y_i = f(\boldsymbol{x}_i)$, find an approximation to the unknown function f, such that it exactly matches the given data-set, i.e. $y_i = \bar{f}(\boldsymbol{x}_i)$. The solution should be applicable to higher dimensions (multivariate) and possess the ability to accommodate large data-sets (large N).

Additionally, in the context of this work, a solution to the scattered data integration problem is also sought. From the same data-set, $\{(\boldsymbol{x}_i,\, y_i)\}_{i=1}^{N}$ with underlying function $y_i = f(\boldsymbol{x}_i)$, find an approximation to integral function $\bar{F}(\boldsymbol{z}) \approx F(\boldsymbol{z}) = \int \cdots \int f(\boldsymbol{x}) d\boldsymbol{\zeta}$, where $\boldsymbol{z}$ and $\boldsymbol{\zeta}$ are partitions of the element $\boldsymbol{x} \in \mathbb{R}^d$. If $\boldsymbol{z} = \emptyset$ and $\boldsymbol{\zeta} \in \mathbb{R}^d$ then the total integral is taken.

It is desired that both $\bar{f}$ and $\bar{F}$ be smooth functions over their entire domains. These problems can be found in many engineering applications, including: Verification and Validation [38, 51], Entry, Descent, and Landing [52], Impact Prediction [41, 39], and pharmacokinetics [43, 53].

Additionally, it should be noted, that the notation and assumptions from the previous section for certain variables (e.g. f, F, $\bar{F}$) are not reflected in this section.

Binning Method

The problem stated above becomes trivial if the data-sites are structured into a linearly-spaced grid. With a gridded structure, an approximation of the underlying function within the domain of the grid may be achieved through linear interpolation of the surrounding grid points. Additionally, the integration of dimensions may be approximated by taking an arithmetic mean across those dimensions of the grid. The total integration of the function may be approximated by the mean of the function values, y_i, scaled by the volume of the data-site grid. The structured format, while straight-forward and intuitive, is restrictive and does not extend well to higher-dimensions. Considering 10 discretizations across each of the 5 dimensions, producing a total of $10^5 = 100{,}000$ data-sites, increasing the number of discritizations of a *single* dimension adds $10^4 = 10{,}000$ data-sites. Incrementing the number of discretizations across all dimensions adds $11^5 - 10^5 = 61{,}051$ data-sites; the method suffers from the curse of dimensionality.

If interpolating the underlying function, f, is not needed but instead the approximation of an integral function, $\bar{F}$, is desired, then a *binning* method may be employed[20, 52]. The binning method does not require a structure on the data-sites, however, unstructured output–or evaluation–points leads to a computationally costly implementation.

The binning process begins by defining a set of integral evaluation points $\{\boldsymbol{z}_k\}_{k=1}^{N_b}$ where N_b is the number of bins. These values will define the geometric center of each bin. While it is not necessary for the bin centers to be structured on a grid–however it

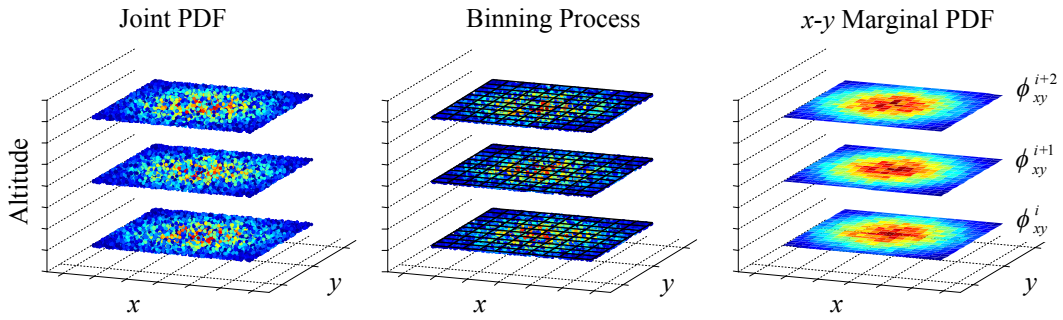


Figure 2.2: Binning method used to marginalize a 5-dimensional joint PDF to two dimensions at three different instances (altitudes)

is the most practical–a method to define the bin boundaries and compute the bin volume is necessary. For scattered bin centers, the bin boundaries may be found using a Voronoi diagram [54] and volume computed using a convex hull algorithm. However, the computational cost of these algorithms become prohibitive with a binning-dimension $p > 3$ and increasing N_b [55]. The computations are trivial for a grid. With the bins defined, the data-sites are projected into the space over which the bins are defined, $\mathbb{R}^p$, where p is the dimension of the bins. The integral value for each bin, $\bar{F}_k(\boldsymbol{z}_k)$, is defined as the mean of the functional values y_i in that bin. The integral values are then normalized by bin volume.

In [20], Leonard et al. used the binning method to marginalize a joint PDF. The five-dimensional, scattered joint PDF data was marginalized to two dimensions and evaluated on a 60×60 grid. Figure 2.2 shows the 5D data projected onto the xy-plane (left), the binning method applied (center), and the resulting 2D marginal PDF after normalizing (right).

While the binning method is straight-forward and easy to implement, its ability to accurately represent the integrated function F clearly depends the on the dimensionality of the problem, the number of data-sites, and the number of bins (the resolution). Consider-

ing a practical implementation, a trade-off arises between coverage (number of data-sites per bin) and the resolution. Additionally, it presents a further problem of how to evaluate the resulting integral function at locations other than the evaluation points; i.e. whether interpolation between points is used, or the bin-center value is constant across the entire bin.

2.2.2 Kernel-Based Methods

Kernel-based methods, also known as meshfree methods, present an attractive solution to the problem presented in Section 2.2.1. The use of Kernel-based methods can be found across various fields in science, engineering, and computer science, including: machine learning applications [56, 57], image processing [58, 59], and path-planning and dynamics problems [60, 19]. The most well-known meshfree method is that of Radial Basis Functions (RBF), covered briefly in Section 2.2.3. The intent of this section is to present the material necessary to understand what kernel-based methods are and how they may be utilized, specifically for the problems covered in this work. Readers interested in a deeper understanding of the material are advised to consult [61], [62], [63], and the references therein.

Formulation

Reiterating the problem statement, kernel-based methods aim to approximate the relationship of the scattered data-set $\{(\boldsymbol{x}_i,\, y_i)\}_{i=1}^{N}$ where the true relationship $y = f(\boldsymbol{x})$ is known at $\{\boldsymbol{x}_i\}_{i=1}^{N}$ only. The function f is approximated by a linear combination of *basis* functions, $\mathcal{B}_i(\boldsymbol{x})$, from $\mathcal{B} = \{\mathcal{B}_1(\boldsymbol{x}), \ldots, \mathcal{B}_M(\boldsymbol{x})\}$, where M is the dimension of the function space $\mathcal{B}$. To ensure that an interpolant from $\mathcal{B}$ exists, the basis functions can be generated from a *kernel* applied to the data-set [61], i.e. $\mathcal{B} = \{K(\boldsymbol{x},\, \boldsymbol{x}_1), \ldots,\, K(\boldsymbol{x},\, \boldsymbol{x}_N)\}$ where the kernel is $K\colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and now $M = N$, the number of data-sites. The interpolant of the

data-set to the unknown function f may now be written as

$$\bar{f}(\boldsymbol{x}) = \sum_{i=1}^{N} c_i \mathcal{B}_i(\boldsymbol{x}) \tag{2.24a}$$

$$= \sum_{i=1}^{N} c_i K(\boldsymbol{x},\, \boldsymbol{x}_i) \tag{2.24b}$$

where the basis coefficients, $\boldsymbol{c} = [c_1, \ldots,\, c_N]^\intercal$, are computed by solving the linear system,

$$\mathcal{K}\boldsymbol{c} = \boldsymbol{y} \tag{2.25}$$

with $(\mathcal{K})_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ $i,\, j = 1, \ldots,\, N$ and $\boldsymbol{y} = [y_1, \ldots,\, y_N]^\intercal$. The kernel matrix, $\mathcal{K}$, is $N \times N$ and a unique solution to Eq. (2.25) exists when $\det \mathcal{K} \neq 0$; it is nonsingular. The kernel K is restricted such that it generates a symmetric, positive definite kernel matrix $\mathcal{K}$, i.e.

$$\boldsymbol{c}^\intercal \mathcal{K} \boldsymbol{c} > 0 \qquad \forall\, \boldsymbol{c} \in \mathbb{R}^N,\ \boldsymbol{c} \neq 0 \tag{2.26}$$

Under this restriction, the kernel matrix will always be nonsingular. Kernels that satisfy this requirement are also known as positive definite [62, 61]. Additionally, if the kernel is also compactly supported then it may generate a sparse matrix, which is beneficial when solving Eq. (2.25) numerically.

Computing Basis Coefficients

With the kernel matrix guaranteed to be nonsingular, Eq. (2.25) may be solved using a variety of methods. Computational resources (mainly memory) permitting, $\boldsymbol{c}$ may be computed using a direct method, such as LU factorization, or, given the restrictions above, Cholesky factorization [64]. However, for large data-sets, an indirect, or iterative, method may be needed to solve the linear system of equations. Given the restrictions on $\mathcal{K}$, symmetric and positive definite, a conjugate gradient scheme may be used.

Derived as a direct method, conjugate gradient (CG) seeks to minimize the residual

error over a growing Krylov subspace at each iteration [65, 66, 67, 64]. In doing so, the method produces an optimal direction and step-size to be taken from the current iterate. Theoretically, the CG method will converge to the exact solution (zero residual error) in k steps, where $k \leq N$ is number of unique eigenvectors of $\mathcal{K}$. When used as an iterative method with an acceptable residual-error tolerance $\|\mathcal{K}\boldsymbol{c}^* - \boldsymbol{y}\| < \epsilon$, a solution may be found in less than k iterations. It is common for the tolerance ϵ to be defined as a relative residual, $\epsilon = \eta\|\boldsymbol{y}\|$ [65]. An additional benefit of the CG method is that it never explicitly requires the matrix $\mathcal{K}$ to be known. Rather, all that is required for a numerical implementation is the result of a matrix-vector product, $w(\boldsymbol{x}) = \mathcal{K}\,\boldsymbol{x}$, known as *matrix free* evaluation. This is of great benefit when $\mathcal{K}$ is sparse. The algorithm to fully implement the method may be found in [65].

2.2.3 Radial Basis Functions

A well known class of positive definite functions are known as Radial Basis Functions (RBF) [68, 66, 63, 62]. These corresponding kernels are of the form,

$$K_R(\boldsymbol{x}, \boldsymbol{x}_i) = \phi\left(\|\boldsymbol{x} - \boldsymbol{x}_i\|\right) \tag{2.27}$$

where $\|\cdot\|$ is the Euclidean norm and $\phi(r)\colon [0,\, \infty) \to \mathbb{R}^+$ is the radial function. Table 2.1 lists three types of commonly used RBFs. For each function, it is also common to add a shape parameter $\alpha > 0$ in the form of $\phi(\alpha r)$, the effects of which are shown in Figure 2.3.

Table 2.1: Common Radial Basis Functions

Name	Function
Gaussian	$\phi(r) = \mathrm{e}^{-r^2}$
Inverse Multiquadratic (IMQ)	$\phi(r) = \frac{1}{\sqrt{1+r^2}}$
Wendland W2	$\phi_{3,2}(r) = (1-r)_+^4(4r+1)$
Wendland W4	$\phi_{3,4}(r) = (1-r)_+^6(35r^2+18r+3)$

The notation of the Wendland functions in Table 2.1 is $\phi_{d_{\max},\,k}$, where $d_{\max}$ is the maxi-

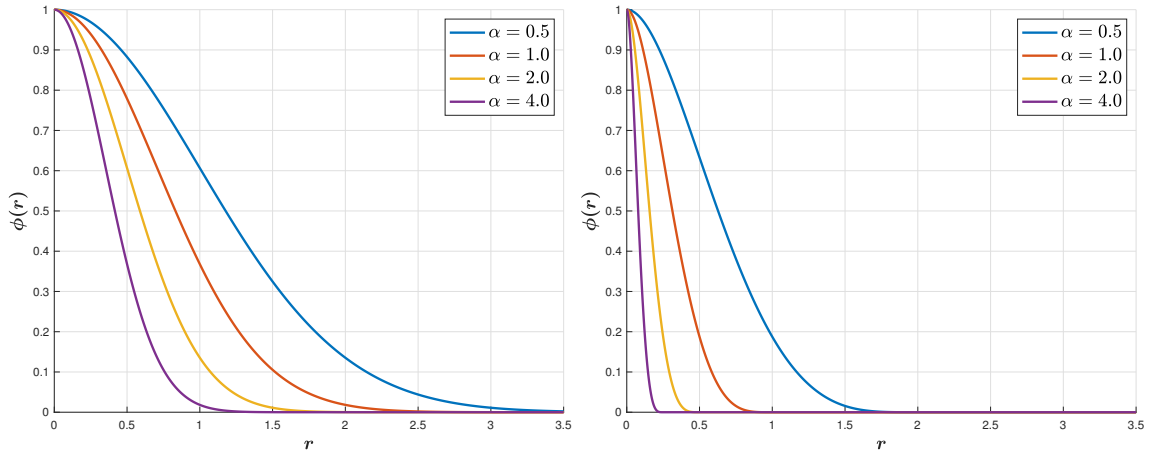


Figure 2.3: Effect of the shape parameter α on Gaussian (left) and Wendland $\phi_{3,2}$ (right) RBFs

mum dimension of the data-set that the function may be applied to and k is the smoothness of the function, C^k. Additionally, $(\cdot)_+$ is a cutoff function equivalent to $\max(\cdot, 0)$.

While there are numerous other RBFs that may be used, the functions in Table 2.1 were chosen to highlight some limitations of using radial functions. The first deals with the support of the functions. While all four functions approach zero as $r \to \infty$, the Gaussian and IMQ functions never *are* zero and are thus not compactly supported. These functions will generate a dense–as opposed to sparse–kernel matrix which will require the use of a direct method to solve Eq. (2.25), effectively limiting the number of data-sites that may be used. In contrast, the Wendland functions are identically zero for $r \geq 1$. It is common to use the shape parameter α, shown in Figure 2.3, to control the support of the functions via the relation $r < 1/\alpha$.

The second limitation of RBFs is the difficulty associated with integrating all-or-some dimensions of the interpolated function. In [69], the total integration of the unknown function is investigated. The authors find that 1.) the compactly supported Wendland W2 and the thin-plate spline (a *conditionally* positive-definite function not shown) are the most reliable and robust and 2.) the ability to successfully solve the problem is correlated to both the number of data-sites and RBF used. As alluded to in Section 2.2.2, the use of compactly

support RBFs (kernels) improves the tractability of a solution for larger data-sets (hundreds of thousands data-sites), this is reaffirmed by [69].

As a final note to the integration of RBFs, the work in [69] does not explicitly address the idea of integrating single, or multiple but not all, dimensions of the interpolant. For scattered data in the unit hypercube $[0,\, 1]^d$, it is shown that the Gaussian RBF may be decomposed using separation of variables, i.e.

$$\phi(r) = \mathrm{e}^{-r^2} = \mathrm{e}^{-\left((x_1 - x_{i,1})^2 + \ldots + (x_d - x_{i,d})^2\right)} \tag{2.28}$$

and the integral of each basis function computed along the individual dimensions independently using the error function,

$$I_i \propto \big(\mathrm{erf}(1 - x_{i,1}) - \mathrm{erf}(-x_{i,1})\big) \cdot \ldots \cdot \big(\mathrm{erf}(1 - x_{i,d}) - \mathrm{erf}(-x_{i,d})\big) \tag{2.29}$$

where the scaling of the solution would depend on the dimension d and, possibly, the shape parameter α. Using this methodology, it would be possible to integrate dimensions independently. However, this separation is typically not possible with radial functions.

2.2.4 Lobachevsky Splines

Lobachevsky splines for use in scattered data interpolation were first introduced in 2010 by Roberto Cavoretto in his PhD thesis, *Meshfree approximation methods, algorithms and applications* [70]. The splines were investigated further in [71, 72, 73], with their primary application being landmark-based image registration [74, 59]. In [59], the performance of the Lobachevsky splines is compared against that of RBFs, including the Gaussian and Wendland W2 functions. In this, and related work by Leonard et al., the splines have been found to be of great use for marginalizing joint PDFs or computing conditional expectations on scattered data [75, 76].

From the derivations provided in [70, 71], the univariate Lobachevsky spline is defined

as

$$f_n^*(\alpha x) = \sqrt{\frac{n}{3}} \frac{1}{2^n (n-1)!} \sum_{k=0}^{n} (-1)^k \binom{n}{k} \left[\sqrt{\frac{n}{3}} \alpha x + (n-2k) \right]_+^{n-1} \qquad \forall x \in \mathbb{R} \quad (2.30)$$

where $\alpha > 0$ is a shape parameter and $n \geq 2$ is an even integer defining the function's smoothness, $\mathcal{C}^n$ (similar to the Wendland functions in Table 2.1). It is shown in [71] that the univariate Lobachevsky spline in Eq. (2.30) is positive definite. Extending to higher dimensions, the multivariate Lobachevsky kernel is the product of the univariate functions, i.e.

$$K_L(\boldsymbol{x}, \boldsymbol{x}_i) = \prod_{h=1}^{d} f_n^* \left(\alpha_h \left(x_h - x_{i,h} \right) \right) \qquad \forall \boldsymbol{x}, \boldsymbol{x}_i \in \mathbb{R}^d \quad (2.31)$$

where α_h is the shape parameter for the h^{th} dimension. In [61], this form of kernel is called a tensor product and from the properties of positive definite functions given by [61, 62], the product of positive definite functions is also positive definite. As such, the Lobachevsky kernel of Eq. (2.31) will generate a symmetric, positive definite kernel matrix $\mathcal{K}$. Figure 2.4 shows the effects of various n and α values on Eq. (2.30).

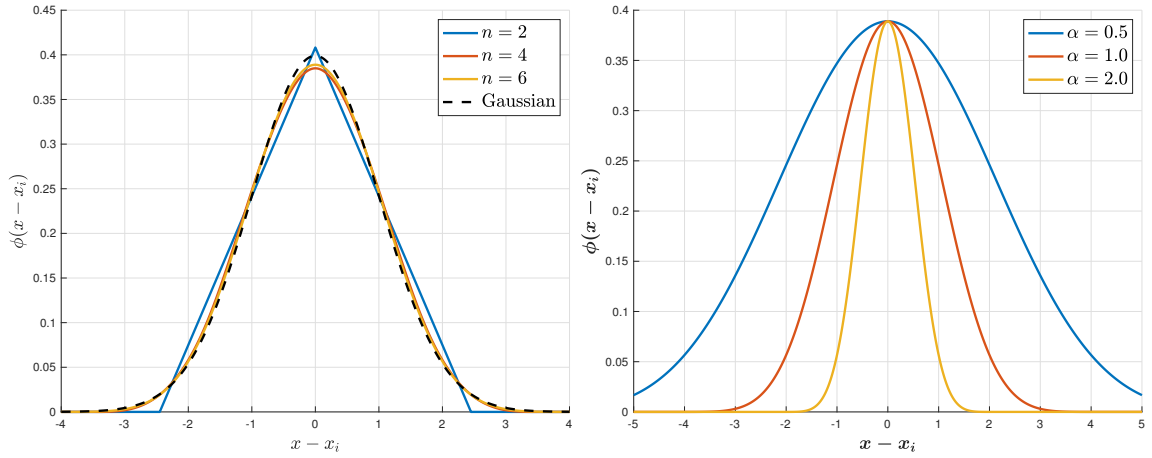


Figure 2.4: The Lobachevsky spline function. (left) The effect of smoothness parameter n (with $\alpha = 1.0$). (right) Effect of varying the shape parameter α on L4 function ($n = 4$)

The Lobachevsky spline possesses a number of unique properties that are of benefit to the problem of interest. First, in the limit as $n \to \infty$ the spline function f_n^* converges to

the standard normal distribution uniformly for all $x \in \mathbb{R}$, this convergence extends to the multivariate kernel [71]. As such, K_L asymptotically behaves like the Gaussian RBF from Section 2.2.3, though it is not radial itself [73]. Next, for finite n, the kernel is compactly supported on the d-dimensional box defined by $[-\sqrt{3n}/\alpha_h,\, \sqrt{3n}/\alpha_h]$ $h = 1, \ldots, d$. In turn, K_L can generate a sparse kernel matrix; a great benefit when implementing numerically. Finally, and maybe most importantly, the Lobachevsky kernel may be integrated analytically along any dimension or combination of dimensions. While all kernel-based interpolants may be integrated via the integration of their basis functions through the linearity of Eq. (2.24), i.e.

$$\int_\Omega \bar{f}(\boldsymbol{x})\, d\boldsymbol{\zeta} = \int_\Omega \sum_{i=1}^{N} c_i K(\boldsymbol{x},\, \boldsymbol{x}_i) d\boldsymbol{\zeta} \tag{2.32a}$$

$$= \sum_{i=1}^{N} c_i \int_\Omega K(\boldsymbol{x},\, \boldsymbol{x}_i) d\boldsymbol{\zeta} \tag{2.32b}$$

where $\boldsymbol{\zeta} \in \Omega \subseteq \mathbb{R}^d$ is the subset of dimensions to be integrated, the Lobachevsky kernel allows the integral(s) of Eq. (2.32b) to be evaluated in closed-form.

The integration process of the Lobachevsky spline and of its multivariate kernel for $d \geq 2$ is investigated in [72] and [73], respectively. Considering the univariate Lobachevsky spline, its integral on the domain $\Omega = [a,\, b]$ is

$$I_\Omega(x_i,\, \alpha) = \int_a^b f_n^*(\alpha(x - x_i))dx = \frac{1}{\alpha}\left[\Phi_n^*(\alpha(b - x_i)) - \Phi_n^*(\alpha(a - x_i))\right] \tag{2.33}$$

with the function $\Phi_n^*(x)$ $x \in \mathbb{R}$ being

$$\Phi_n^*(x) = \frac{1}{2^n n!} \sum_{k=0}^{n} (-1)^k \binom{n}{k} \left[\sqrt{\frac{n}{3}}x + (n - 2k)\right]_+^n \tag{2.34}$$

which converges to the standard normal CDF as $n \to \infty$. As such, in the limit, the integral of Eq. (2.33) asymptotically approaches that of Eq. (2.29).

Using the kernel-based method of Section 2.2.2, the Lobachevsky interpolant of the unknown function f, written out explicitly, is,

$$\bar{f}(\boldsymbol{x}) = \sum_{i=1}^{N} c_i K_L(\boldsymbol{x},\, \boldsymbol{x}_i) = \sum_{i=1}^{N} c_i \prod_{h=1}^{d} f_n^*(\alpha(x_h - x_{i,h})) \tag{2.35a}$$

$$= \sum_{i=1}^{N} c_i f_n^*(\alpha(x_1 - x_{i,1})) \cdot \ldots \cdot f_n^*(\alpha(x_d - x_{i,d})) \tag{2.35b}$$

with the basis coefficients, $\boldsymbol{c}$, computed by solving $\mathcal{K}\boldsymbol{c} = \boldsymbol{y}$. For notational simplicity, α is assumed to be constant across all dimensions. Leveraging the structure of Eq. (2.35b), the integral of any dimension, or dimensions, may be analytically evaluated. The element $\boldsymbol{x} \in \mathbb{R}^d$ is partitioned into $\boldsymbol{\zeta} \in \mathbb{R}^m$ and $\boldsymbol{z} \in \mathbb{R}^{d-m}$, the *integrated* and *remaining* dimensions, respectively, such that $\boldsymbol{z} \cup \boldsymbol{\zeta} = \boldsymbol{x}$ and $\boldsymbol{z} \cap \boldsymbol{\zeta} = \emptyset$.

$$\bar{F}(\boldsymbol{z}) = \int \cdots \int_{\Omega_{\boldsymbol{\zeta}}} \bar{f}(\boldsymbol{x}) d\boldsymbol{\zeta} \tag{2.36a}$$

$$= \int \cdots \int_{\Omega_{\boldsymbol{\zeta}}} \sum_{i=1}^{N} c_i f_n^*(\alpha(x_1 - x_{i,1})) \cdot \ldots \cdot f_n^*(\alpha(x_d - x_{i,d})) d\zeta_1 \ldots d\zeta_m \tag{2.36b}$$

$$= \sum_{i=1}^{N} c_i f_n^*(\alpha(z_1 - z_{i,1})) \cdot \ldots \cdot f_n^*(\alpha(z_{d-m} - z_{i,d-m})) \times \int_{\Omega_{\boldsymbol{\zeta}}^{(1)}} f_n^*(\alpha(\zeta_1 - \zeta_{i,1})) d\zeta_1 \cdot \ldots \cdot \int_{\Omega_{\boldsymbol{\zeta}}^{(m)}} f_n^*(\alpha(\zeta_m - \zeta_{i,m})) d\zeta_m \tag{2.36c}$$

and using the univariate integral defined in Eq. (2.33),

$$\bar{F}(\boldsymbol{z}) = \sum_{i=1}^{N} c_i \prod_{h=1}^{d-m} f_n^*(\alpha(z_h - z_{i,h})) \times \prod_{h=1}^{m} I_{\Omega_{\boldsymbol{\zeta}}^{(h)}}(\zeta_{i,h},\, \alpha) \tag{2.37a}$$

$$= \sum_{i=1}^{N} \tilde{c}_i \prod_{h=1}^{d-m} f_n^*(\alpha(z_h - z_{i,h})) \tag{2.37b}$$

where the domain of integration for dimension h, $\Omega_{\boldsymbol{\zeta}}^{(h)}$, is defined by the minimum and maximum values of the data-set in that dimension. Since the integral function in Eq. (2.37a)

does not depend on the evaluation point $\boldsymbol{z}$, the contributions from each data-site's $I(\cdot)$ may be multiplied into its c_i, yielding the new set of coefficients $\tilde{\boldsymbol{c}}$. From Eq. (2.37b), $\bar{F}$ constitutes the *analytical* integration of the *approximated* function $\bar{f}$ across the chosen dimensions. As such, during the integration process the interpolation errors, $e(\boldsymbol{x}) = \bar{f}(\boldsymbol{x}) - f(\boldsymbol{x})$, are aggregated. This circumstance is investigated in [73], however, the authors only consider the full-integration case.

2.2.5 Example

Kernel-based approximation methods, while a strong solution to the scattered data interpolation and integration problem, present challenges when considering high-dimensions and large data-sets. For use in this work, a C++ library named InterpCL was created to implement the algorithms.

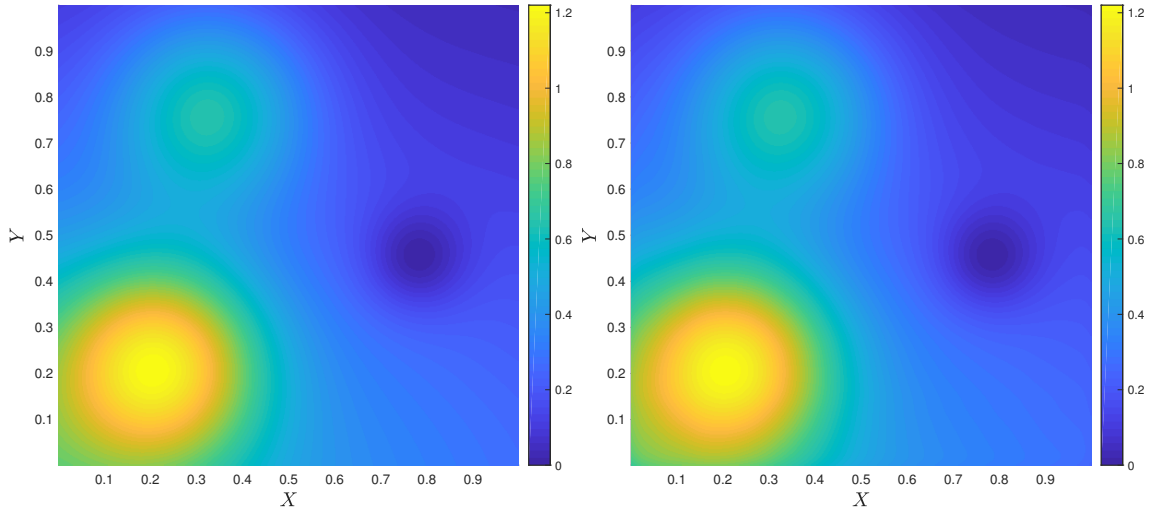
To handle potentially large, high-dimensional data-sets, the library targets graphical processing units (GPU) using the OpenCL standard [77]. Using object-oriented programming (OOP) and the polymorphism that it affords [78], InterpCL was designed to provide a simple interface through which to apply the kernel-based methods, including interpolation, evaluation, and possibly integration. Targeting data-sets in the range of hundreds of thousands of data-sites, InterpCL implements the conjugate gradient (CG) solver discussed in Section 2.2.2. As such, it is designed for sparse kernel matrices, like those generated by the the Lobachevsky and Wendland kernels, where matrix-free methods allow for the tractable evaluation of the matrix-vector products. However, the class structure of the library provides the means for a direct solver to be easily implemented which could extend the library to globally supported kernels.

To highlight the capabilities of the library, an example is presented. The function of interest is five-dimensional, $f\colon \mathbb{R}^5 \to \mathbb{R}$ with dimensions $\boldsymbol{x} = [x,\, y,\, p_1,\, p_2,\, p_3]^\intercal$ and is

defined as

$$f(\boldsymbol{x}) = \text{franke}(x,\, y) \times P(p_1) \times P(p_2) \times P(p_3) \tag{2.38}$$
$$P(z) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)$$

where franke is Franke's function [79], a well-known test function for scattered data interpolation (shown in Figure 2.5a), and P is the PDF of the normal distribution with $\mu = 0.5$ and $\sigma = 0.5/3.5$. To create the data-set, $N = 512{,}000$ data-sites were generated using a Halton sequence [80], filling the $[0, 1]^5$ hypercube, and Eq. (2.38) evaluated at each.



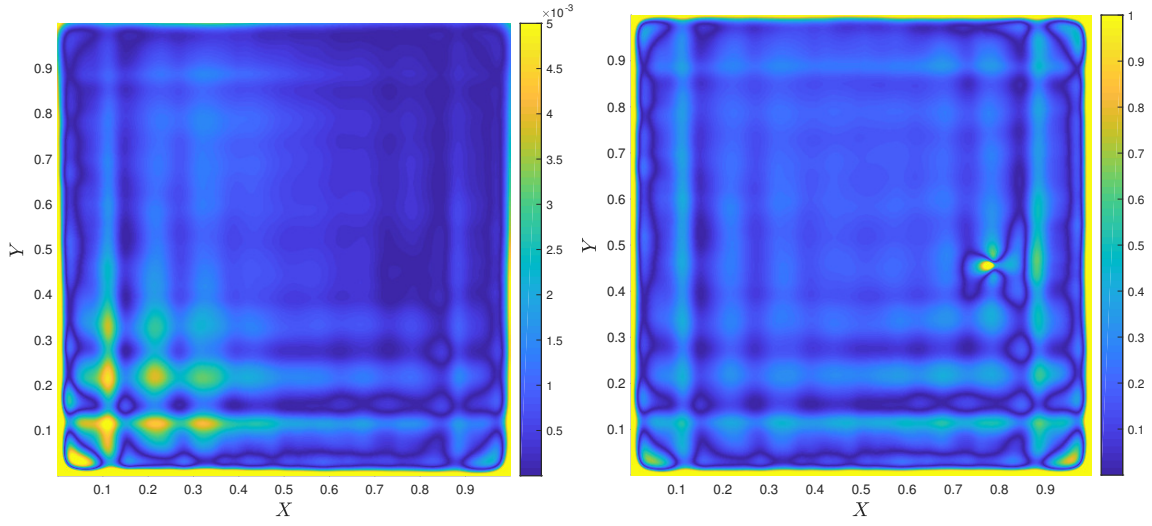
(a) Franke's function (b) Recovered function

Figure 2.5: Comparison of the true Franke's function (left) and recovered function using Lobachevsky spline integration (right)

To recover the underlying function, the 5D data-set is first interpolated using an L4 (Lobachevsky $n = 4$) spline. The shape parameter is chosen such that the compact support is ± 0.22 from the center of each data-site in each dimension. This creates a $512{,}000 \times 512{,}000$ kernel matrix that is 99% sparse. The basis coefficients are computed using the CG solver, terminating at a relative-residual error of 1e^{-8}. For this case, the solver required 932 iterations. Accelerated by a NVIDIA K40 GPU, the process takes 1 hour 49 minutes to complete. The resulting interpolant is evaluated at $256{,}000$ points and, when compared

to the true function value, produces a root-mean-squared(RMS) error of 0.08. However, it should be noted that the maximum function value in $[0,\ 1]^5$ is approximately 25.0.

Following the integration process presented in Section 2.2.4, the interpolant is integrated along the p_1, p_2, and p_3 dimensions. Figure 2.5 shows the recovered function–an approximation of the underlying function–evaluated on a 500×500 grid. It can be seen that the kernel-based method produces a smooth, continuous function. The absolute error between the recovered function and the true Franke's function is shown in Figure 2.6a. Additionally, the relative error–absolute error divided by the true Franke's function–is shown in Figure 2.6b. The RMS error is computed to be $2\mathrm{e}^{-3}$ with a maximum function value of 1.22.



(a) Absolute error (b) Relative error

Figure 2.6: (left) The absolute error between the recovered and true functions. The plot is saturated at $5\mathrm{e}^{-3}$ to mitigate large edge errors. (right) The relative error, saturated at 1%.

In Figures 2.6a and 2.6b it can be seen that the greatest error arises on the lower and left-hand side borders. These errors may be–at least partially–attributed to Franke's function, and therefore the full function in Eq. (2.38), not being a member of the kernel's *native space* [61]. Considering the basis generated by the compactly supported Lobachevsky splines, its native space will be of compactly supported functions, which also have compactly supported derivatives. Franke's function is the composition of two Gaussian peaks,

a Gaussian dip, and a sloping surface. The Gaussian elements, and their derivatives, while not compactly supported, do asymptotically approach zero. However, the sloping surface is not compactly supported, nor does its derivatives go to zero. As such, it will be a source of large edge error [61]. Additionally, kernel-based methods are subject to Gibbs phenomenon [81], which can be seen moving inward from the edges in Figure 2.6b.

InterpCL, leveraging the Lobachevsky splines, will be employed later in this work to perform the computation of conditional expectations in conjunction with the Koopman operator. Fortunately, as discussed in Section 2.1.2, the infinitesimal Koopman operator requires the objective function to be compactly supported, helping to eliminate the large edge error shown above.

CHAPTER 3

CONTROL OPTIMIZATION UNDER UNCERTAINTY

The goal of this work is to propose a probabilistic control optimization methodology leveraging the Frobenius-Perron and Koopman operators that is applicable to the precision airdrop problem. However, the methodology of this section is developed in a generalized sense as it may be applied to wide range of optimization problems.

3.1 Generalized Optimization Problem

To begin formulating the general optimization problem, it is important to separate–and define–the notion of a *decision*-space and an *objective*-space. The decision-space is considered to be the space, or subspace, in which all initial conditions may be found. Put another way, the decision-space is defined as

$$\mathcal{D} = \{f_0(\boldsymbol{x}) > 0,\ t = t_0 \colon\ \boldsymbol{x} \in \Omega,\, t \in \mathbb{R}\} \tag{3.1}$$

where f_0 is the joint PDF describing the initial condition uncertainty at an arbitrary time t_0. The objective-space is the space over which the objective function–a measure of success or cost–is to be applied and can be defined as

$$\mathcal{O} = \{h(t,\, \boldsymbol{x}) = 0,\ t \geq t_0 \colon\ \boldsymbol{x} \in \Omega,\, t \in \mathbb{R}\} \tag{3.2}$$

where h is an arbitrary function of the state-space and, possibly, time. The function h may for instance be defined at some terminal condition of the system, signifying the end-point of any possible system trajectory. As they are defined, the decision and objective-spaces may have spatial and temporal components and as such $\mathcal{D},\, \mathcal{O} \in\ \Omega \times \mathbb{R}$. The spaces are

connected through the system dynamics and the mapping is therefore $\mathrm{S}\colon \mathcal{D} \to \mathcal{O}$.

The optimization problem of interest is one in which an objective function–applied over the objective-space–is maximized. The use of an objective, or score, over a cost allows for the function to be compactly supported. Given the uncertainty in the initial conditions, and therefore uncertainty supported on the objective-space, the objective function can be evaluated probabilistically using an expected value. It follows that the *expected value* of the objective function is, explicitly, the function to be optimized, i.e.

$$\boldsymbol{u}^* = \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{O}}\left[g(\boldsymbol{X}) \mid \boldsymbol{u}\right] \tag{3.3}$$

where $\boldsymbol{u}^*$ is the optimal control decision, $\mathcal{U}$ is the domain of acceptable control decisions, and $\mathbb{E}_{\mathcal{O}}[\cdot]$ is the expected value of the objective function g supported on the objective-space $\mathcal{O}$. The expected value is evaluated considering that $X \sim f_{\mathcal{O}}$, the joint PDF supported on the objective space.

The control decision variable, noted as $\boldsymbol{u}$, can enter the optimization problem in two different ways: 1) through the initial condition uncertainty, $f_0 = f(\boldsymbol{x} \mid \boldsymbol{u})$, and/or 2) through the system dynamics, $\mathrm{S}_t = \mathrm{S}(\boldsymbol{x}, \boldsymbol{u})$. As an example, consider a damped oscillator with Gaussian uncertainty in the initial position and velocity of the system. If the user may change the mean and/or variance of the initial conditions, they are in essence searching for the optimal density in a set. On the other hand, if the user may not change the PDF of the initial conditions but may pick a damping coefficient (on which the system response is parametrized), they are searching for the optimal system in a set characterized by a control variable, the damping coefficient.

Equation (3.3) represents the probabilistically optimal control selection given an objective function and taking into consideration initial condition uncertainty. The difference between its solution, and a *deterministic* solution (when the initial condition uncertainty is neglected) depends on the relationship between the two elements of the expected value:

the forward propagated uncertainty and the objective function. Within the support of the PDF quantifying the forward propagated uncertainty, the difference is driven by the level of symmetry of both the objective about its maximal point and the PDF itself. In the case of a symmetric objective function and PDF, the deterministic and probabilistic solutions will match. Additionally, given a PDF with very narrow support or an objective function with a very shallow gradient (about its maximal point) the differences will become negligible. Therefore, at the discretion of the user, for problems with minimal uncertainty (narrow PDF support), a symmetric objective function, or a *flat* objective function within the PDF's support, a deterministic solution may be sufficient.

3.2 Utilizing Frobenius-Perron and Koopman Operators

The optimized function of Eq. (3.3) is the expected value of the objective function with respect to the joint PDF when it is supported on the objective-space. Written using the inner product notation, it is equivalent to

$$\mathbb{E}_{\mathcal{O}}\left[g(\boldsymbol{X}) \mid \boldsymbol{u}\right] = \int_{\Omega} f_{\mathcal{O}}(\boldsymbol{x} \mid \boldsymbol{u}) g(\boldsymbol{x})\, d\boldsymbol{x} = \langle f_{\mathcal{O}},\, g \rangle \tag{3.4}$$

Employing the Frobenius-Perron operator, the joint PDF supported on the objective-space may be written as a transformation of the initial condition uncertainty, i.e, $f_{\mathcal{O}}(\boldsymbol{x} \mid \boldsymbol{u}) = \mathrm{P_S} f_0(\boldsymbol{x} \mid \boldsymbol{u})$, where $\boldsymbol{u}$ may also be found in S and therefore $\mathrm{P_S}$. Substituting the quantity into Eq. (3.4),

$$\mathbb{E}_{\mathcal{O}}\left[g(\boldsymbol{X}) \mid \boldsymbol{u}\right] = \langle \mathrm{P_S} f_0,\, g \rangle \tag{3.5}$$

The expected value is a function of f_0 and S, both of which may contain the control decision $\boldsymbol{u}$.

Leveraging the adjoint relation from Section 2.1.3 and Eq. (2.17), an equivalent ex-

pected value is expressed using the Koopman operator,

$$\begin{aligned}\langle \mathrm{P}_{\mathrm{S}} f_0,\, g\rangle &= \langle f_0,\, U_{\mathrm{S}} g\rangle \\ &= \int_{\Omega} f_0(\boldsymbol{x} \mid \boldsymbol{u})\, U_{\mathrm{S}} g(\boldsymbol{x})\, d\boldsymbol{x} \qquad (3.6)\\ &= \mathbb{E}_{\mathcal{D}}\left[U_{\mathrm{S}} g(\boldsymbol{X}) \mid \boldsymbol{u}\right] \qquad (3.7)\end{aligned}$$

where $\mathbb{E}_{\mathcal{D}}[\cdot]$ is evaluated over the decision-space with $\boldsymbol{X} \sim f_0$. Considering the optimization problem of Eq. (3.3) to be the Frobenius-Perron form, an alternative, but equivalent, problem may be written using the Koopman operator

$$\boldsymbol{u}^* = \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{D}}\left[U_{\mathrm{S}} g(\boldsymbol{X}) \mid \boldsymbol{u}\right] \qquad (3.8)$$

where the optimized function is still explicitly dependent on $\boldsymbol{u}$ through its appearance in f_0 and/or S.

The choice of constructing the control optimization problem using the Koopman operator versus the Frobenius-Perron operator, and vice versa, is problem dependent. Each methods brings associated benefits, costs, and restrictions. Considering a problem where the system dynamics S, or F, are not a function of the control selection $\boldsymbol{u}$, the Koopman operator approach provides a method to pull-back the objective function *once* and then compute the expected value for each $f_0(\boldsymbol{x} \mid \boldsymbol{u})$ directly on the decision-space. However, when using the Koopman operator, the joint PDF supported on the objective-space is not needed nor explicitly computed. If this forward-propagated density is of value when solving the control optimization problem, the Frobenius-Perron operator approach may be used. Even so, care must be taken when evaluating the forward propagated joint PDF numerically using Eq. (2.11). Given a strictly positive or strictly negative $\Phi(\boldsymbol{x})\ \ \forall\, \boldsymbol{x} \in \Omega$, the joint PDF will approach 0 or ∞, respectively, given enough time, causing floating-point precision errors.

3.3 Inequality Constraints

The optimization problems of Eqs. (3.3) and (3.8) may be easily extended to include inequality constraints over the augmented state space. Given the uncertainty that exists in the initial conditions, any constraint is inherently probabilistic and expressed as an expected value. Constructed in a similar fashion to the decision and objective spaces, a constraint space is expressed as

$$\mathcal{C} = \left\{ \left| q(t,\, \boldsymbol{x}) \right| > 0 \colon \boldsymbol{x} \in \Omega,\ t \in \mathbb{R} \right\} \tag{3.9}$$

where q is the constraint function. Considering the semidynamical system $\{\mathrm{S}_t\}_{t\geq 0}$, the mapping $\mathrm{Q}\colon \mathcal{D} \to \mathcal{C}$, $\mathrm{Q} \in \{\mathrm{S}_t\}_{t\geq 0}$ connects the decision-space to the constraint space where Q is indexed from the set via the parameter t_q. If t_0 indexes the decision-space and t_f the objective-space, it is assumed that $t_0 \leq t_q \leq t_f$. The generality of the definition allows for the inclusion of constraints that are: 1) supported on the decision-space, $\mathcal{C} \subseteq \mathcal{D}$ with $t_q = t_0$, 2) supported on the objective-space, $\mathcal{C} \subseteq \mathcal{O}$ with $t_q = t_f$, or 3) supported anywhere in between, i.e, $t_0 < t_q < t_f$.

Considering the mapping Q, the Frobenius-Perron operator may be used to push-forward the initial condition PDF to where it is supported on the constraint-space for use in an expected value calculation. The constrained optimization problem in the Frobenius-Perron form is written as

$$\boldsymbol{u}^* = \underset{\boldsymbol{u}\in\mathcal{U}}{\arg\max}\, \mathbb{E}_{\mathcal{O}} \left[g(\boldsymbol{X}) \mid \boldsymbol{u} \right], \qquad \boldsymbol{X} \sim \mathrm{P}_{\mathrm{S}} f_0 \tag{3.10a}$$

$$\text{s.t} \quad \mathbb{E}_{\mathcal{C}} \left[q(\boldsymbol{X}) \mid \boldsymbol{u} \right] \leq \lambda, \qquad \boldsymbol{X} \sim \mathrm{P}_{\mathrm{Q}} f_0 \tag{3.10b}$$

where λ is the constraint tolerance. Alternatively, and equivalently, the Koopman operator may be used to pull-back the constraint function from the constraint-space to the decision-

space. The constrained optimization problem in the Koopman form is written as

$$\boldsymbol{u}^* = \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{D}}\left[\mathrm{U_S} g(\boldsymbol{X}) \mid \boldsymbol{u}\right], \qquad \boldsymbol{X} \sim f_0 \tag{3.11a}$$

$$\text{s.t} \quad \mathbb{E}_{\mathcal{D}}\left[\mathrm{U_Q} q(\boldsymbol{X}) \mid \boldsymbol{u}\right] \leq \lambda, \qquad \boldsymbol{X} \sim f_0 \tag{3.11b}$$

where both expected value calculations are performed on the decision-space.

Using this methodology, an arbitrary number of constraints may be considered where q_i, $\mathcal{C}_i$, and λ_i are the equation, space, and tolerance of the i^{th} constraint, respectively.

3.4 Change of Variables

In the sections above, S, and possibly Q, are members of the semidynamical system generated by a set of ODEs with time t as the independent variable, i.e. $\mathrm{S},\, \mathrm{Q} \in \{\mathrm{S}_t\}_{t\geq 0}$. As such, the corresponding Frobenius-Perron and Koopman operator semigroups, $\{\mathrm{P}_t\}_{t\geq 0}$ and $\{\mathrm{U}_t\}_{t\geq 0}$, respectively, are parametrized by time. Additionally, when equipped with an infinitesimal operator, the ODE form from the MOC (Sections 2.1.1 and 2.1.2) is with respect to time. However, when defining the objective-space, or constraint-space(s), it is often with respect to a function of the state-space and not time, i.e. $h(t,\, \boldsymbol{x}) \neq h(t)$. Therefore, the semidynamical system and corresponding semigroups/infinitesimal operators must be *reparametrized* so that a single mapping delivers all possible initial conditions to the space of interest.

The reparametrization is accomplished by first considering the impact of a space not explicitly dependent on time. From the definition of an objective-space, for example, that is a subspace of the state-space, the time of the event t_e (the time needed by an initial condition to reach the space) is defined to be the solution of

$$h(t_e,\, \mathrm{S}_{t_e}(\boldsymbol{x}_0)) = 0 \qquad \forall \boldsymbol{x}_0 \in \mathcal{D} \tag{3.12}$$

where h is from Eq. (3.2). It is assumed that t_e is finite for all $\boldsymbol{x}_0 \in \mathcal{D}$. While two different initial conditions may have the same event time, i.e. $h_e \colon \boldsymbol{x}_0 \mapsto t_e$ is not unique, it is the possibility that h_e maps to something other than a constant for all initial conditions that forces a reparametrization.

To make the mapping above consistent for all $\boldsymbol{x}_0 \in \mathcal{D}$, a new variable v is introduced. It is assumed that $v = \phi(t, \boldsymbol{x})$ and the relationship ϕ satisfies

$$v(0) = \phi(0,\, \boldsymbol{x}) = 0 \qquad \forall \boldsymbol{x}_0 \in \mathcal{D} \tag{3.13a}$$

$$v(t_e) = \phi(t_e,\, \boldsymbol{x}) = v_e \qquad \forall \boldsymbol{x}_0 \in \mathcal{O} \tag{3.13b}$$

$$dv = \dot{\phi}(t,\, \boldsymbol{x})dt \qquad 0 \leq t \leq t_e \tag{3.13c}$$

where, from the definition of the objective–and other–spaces, $t_e \neq 0$ and $v_e \neq 0$. Because, as an independent variable, time was taken to be monotonic, it is assumed that the function ϕ is monotonic for all $0 \leq t \leq t_e$ to ensure that the transformation is invertible. As such, $\dot{\phi}(t,\, \boldsymbol{x}) \neq 0$ for all $0 \leq t \leq t_e$.

Using Eqs. (3.13), the change of variables approach may be applied to the ODEs of the semidynamical system of Eq. (2.6),

$$\boldsymbol{x}_v = \begin{bmatrix} \boldsymbol{x} \\ t \end{bmatrix} \tag{3.14a}$$

$$\frac{d\boldsymbol{x}_v}{dv} = \frac{1}{\dot{\phi}(\boldsymbol{x}_v)} \begin{bmatrix} F(\boldsymbol{x}) \\ 1 \end{bmatrix} \tag{3.14b}$$

where the state-space is augmented to include time *explicitly* as a state, $\boldsymbol{x}_v \in \mathbb{R}^{d+1}$. The necessity of the monotonic condition of the function ϕ can be seen in Eq. (3.14b).

The semigroups of Frobenius-Perron and Koopman operators generated by Eq. (3.14b) may now be expressed as $\{\mathrm{P}_v\}_{0 \leq v \leq v_e}$ and $\{\mathrm{U}_v\}_{0 \leq v \leq v_e}$, respectively, parametrized by the

new variable v. The infinitesimal operators may be derived from each, i.e.

$$\frac{du_{\mathrm{P}}}{dv} = \frac{du_{\mathrm{P}}}{dt}\left(\frac{dt}{dv}\right) = -u(v)\frac{\Phi(\boldsymbol{x}_v)}{\dot{\phi}(\boldsymbol{x}_v)}\bigg|_{\boldsymbol{x}_v(v)} \tag{3.15a}$$

$$\frac{du_{\mathrm{U}}}{dv} = \frac{du_{\mathrm{U}}}{dt}\left(\frac{dt}{dv}\right) = 0\bigg|_{\boldsymbol{x}_v(v)} \tag{3.15b}$$

where, once again, the necessity of a monotonic ϕ may be seen in Eq. (3.15a). However, it is shown in [32] that singularities occurring from $\dot{\phi} = 0$ may be mitigated in certain circumstances.

For the Koopman operator, this change of variables approach is trivial, as shown in Eq. (3.15b). This is due, in part, by the fact that the Koopman operator is the means through which a function of observables is transported through the state-space. As such, the change of variables may be considered an alteration to the function itself and not the underlying state-space. Therefore, the direct relationships ϕ and ϕ' do not need to be explicitly known.

For the Frobenius-Perron operator, where the density f is inherently connected to the state-space–through its integral definition in Eq. (2.4)–the change is not as simple. An explicit form of $\dot{\phi}$ is needed in Eq. (3.15a). In the case where v is another dimension of the state-space, i.e. $v = x_i$, $1 \leq i \leq d$, the derivative relationship is known, $\dot{\phi}(\boldsymbol{x}) = F_i(\boldsymbol{x})$, where F_i is from Eq. (2.6). This case is investigated in [82]. What the authors termed *terrain compensation* may be attributed to the required normalization of Eq. (3.13b) to a single value v_e for all possible $\boldsymbol{x}_0 \in \mathcal{D}$.

3.5 Spring-Mass-Damper Example

This section presents a simple example utilizing the Frobenius-Perron and Koopman operator optimization approaches, highlighting the benefits of the operator-theoretic formulation of control selection under uncertainty. The system of interest is a simple spring-mass-damper (SMD) with an externally applied control force. The equations of motion of the

system are

$$m\ddot{x} + c\dot{x} + kx = F[u(t - t_{start}) - u(t - (t_{start} + \Delta t)] \tag{3.16}$$

where $u(t-a)$ is the unit step function, t_{start} is the time at which the force is applied, and Δt is the duration. The goal is to determine the optimal t_{start} to probabilistically maximize an objective function at $T = 10$s, the final time. As such, the optimal control decision is denoted as $\boldsymbol{u}^*$, which, while it is a scalar value, is left bolded to avoid confusion with the step function. The system parameters used in Eq. (3.16) may be found in Table 3.1.

Table 3.1: Spring-Mass-Damper system parameters

Parameter	Value
Mass (m)	1.0 kg
Spring constant (k)	1.0 N/m
Damping constant (c)	0.3 Ns/m
Force (F)	1.0 N
Applied time (Δt)	0.5 s

To utilize the control optimization procedure developed in the previous sections, the EOM of Eq. (3.16) must be written in state-space notation. Additionally, the state-space EOMs must be time invariant. From Eq. (3.16), it can be seen that the step function has a dependence on time. To satisfy this restriction, two state-space systems are developed, unforced and forced variants, with states $\boldsymbol{x} = [x,\, \dot{x}]^\intercal$

$$\text{Unforced}: \quad \dot{\boldsymbol{x}}_U = A\boldsymbol{x} \qquad = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{3.17a}$$

$$\text{Forced}: \quad \dot{\boldsymbol{x}}_F = A\boldsymbol{x} + \boldsymbol{b} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ F \end{bmatrix} \tag{3.17b}$$

where the instances t_{start} and $t_{start} + \Delta t$ signal the unforced-to-forced and forced-to-unforced switches, respectively. The state-space mapping S_t of the system may be created

by splicing together the unforced and forced stages according to

$$\text{Stage 1:} \quad \boldsymbol{x}_1(t) = \mathrm{e}^{At}\boldsymbol{x}_0 \qquad 0 \le t \le t_s \tag{3.18a}$$

$$\text{Stage 2:} \quad \boldsymbol{x}_2(t) = \mathrm{e}^{A(t-t_s)}\boldsymbol{x}_1(t_s) + \left(\mathrm{e}^{A(t-t_s)} - I\right) A^{-1}\boldsymbol{b} \qquad t_s < t \le t_s + \Delta t \tag{3.18b}$$

$$\text{Stage 3:} \quad \boldsymbol{x}_3(t) = \mathrm{e}^{A(t-(t_s+\Delta t))}\boldsymbol{x}_2(t_s + \Delta t) \qquad t > t_s + \Delta t \tag{3.18c}$$

where t_{start} was shortened to t_s for simplicity. The full mapping $\mathrm{S}_T \colon \boldsymbol{x}_0 \mapsto \boldsymbol{x}_T$ may be written as a single entity by substituting Stage 1 into Stage 2, and the result into Stage 3. Figure 3.1 shows an example trajectory of the mass displacement as a function of time with an initial displacement of 0.3m and the input force applied at $t_{start} = 2$s.

The Frobenius-Perron and Koopman operators of the mapping may be formulated in a similar fashion, respectively,

$$\mathrm{P}_T f = \mathrm{P}_3(\mathrm{P}_2(\mathrm{P}_1 f)) = \mathrm{P}_3\mathrm{P}_2\mathrm{P}_1 f \tag{3.19}$$

$$\mathrm{U}_T g = \mathrm{U}_1(\mathrm{U}_2(\mathrm{U}_3 g)) = \mathrm{U}_1\mathrm{U}_2\mathrm{U}_3 g \tag{3.20}$$

where the number values correspond to the stages in Eqs. (3.18). Note that because the underlying systems differ, Eq. (3.17a) versus Eq. (3.17b), the operators may not be *added* together, e.g. $\mathrm{P}_t\mathrm{P}_{t'} = \mathrm{P}_{t+t'}$.

3.5.1 Initial Condition Uncertainty

For this example, uncertainty in initial position, $x(0)$, and velocity, $\dot{x}(0)$, are considered. The variables are considered random, exhibiting a Uniform distribution over the domain $\Omega = [0,\, 0.5]^2$. It is assumed that there is no uncertainty in the system parameters. The joint PDF is then defined as:

$$f_0 = 4 \quad \forall\, x, \dot{x} \in \Omega \tag{3.21}$$

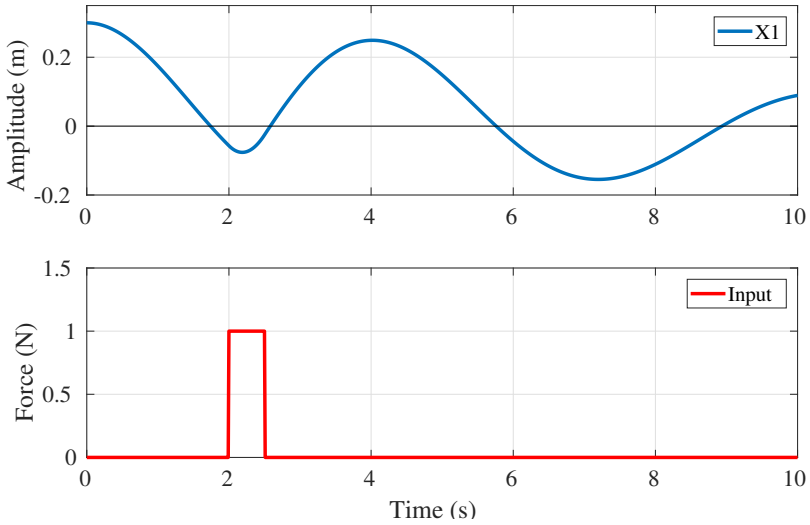


Figure 3.1: Example Trajectory with Input Force at $t_{start} = 2s$

The objective function is defined only over the position at the final time:

$$g(\boldsymbol{x}(T)) = 1 - [x(T)]^2 \tag{3.22}$$

The optimal selection of t_{start}, according to the objective function in Eq. (3.22), may be expressed using the Frobenius-Perron or Koopman operators, discussed in Section 3.2, respectively,

$$\begin{aligned} \boldsymbol{u}^* &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}[g(\boldsymbol{X}) \mid \boldsymbol{u}] \\ &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \iint_{\Omega} (1-x^2)\mathrm{P}_T f_0(x,\, \dot{x}) dx d\dot{x} \end{aligned} \tag{3.23a}$$

$$\begin{aligned} \boldsymbol{u}^* &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}[\mathrm{U}_T g(\boldsymbol{X}) \mid \boldsymbol{u}] \\ &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \iint_{\Omega} \mathrm{U}_T(1-x^2) f_0(x,\, \dot{x}) dx d\dot{x} \end{aligned} \tag{3.23b}$$

where both P_T and U_T are dependent on $\boldsymbol{u}$ (t_{start}). According to the adjoint property, the results of Eqs. (3.23a) and (3.23b) will be identical. Using Eq. (3.23a), the expected value integral may be solved analytically leveraging the system's linearity. Figure 3.2 shows the expected objective function value for each possible control decision, t_{start}.

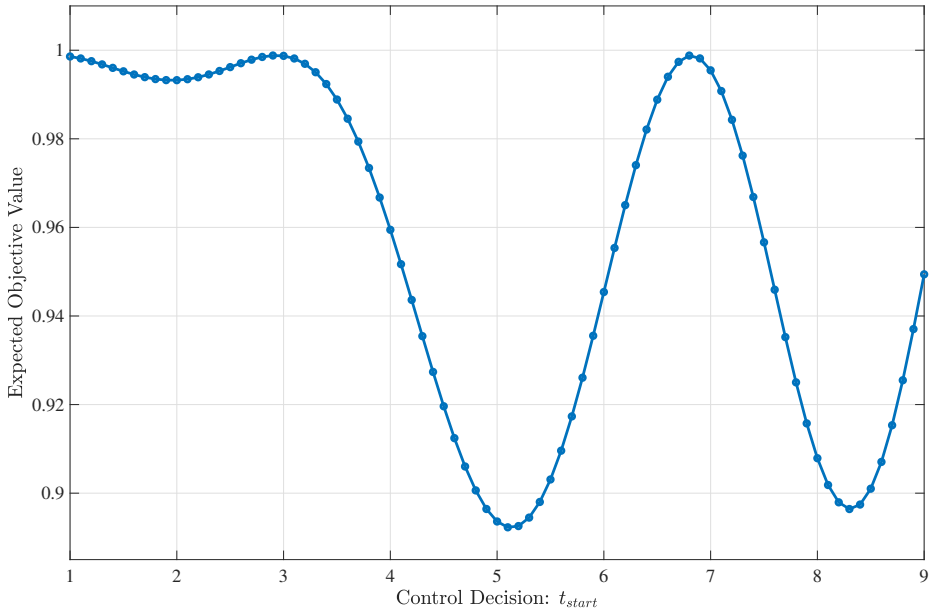


Figure 3.2: Expected objective function value versus possible control decisions, t_{start}

To compare the Frobenius-Perron and Koopman operator approaches to Monte Carlo simulations, the integrals of Eqs. (3.23) are computed numerically using Monte Carlo integration such that there exists a dependence on the number of samples/discretizations for each method. The Monte Carlo, Frobenius-Perron, and Koopman approaches are given, respectively, as

$$\mathbb{E}[g(\boldsymbol{X} \mid \boldsymbol{u})] \approx \frac{1}{N} \sum_{i=1}^{N} g(\boldsymbol{x}_i(T)) \tag{3.24a}$$

$$\mathbb{E}[g(\boldsymbol{X}) \mid \boldsymbol{u}] \approx V(T) \frac{1}{N} \sum_{i=1}^{N} g(\boldsymbol{x}_i(T)) \, \mathrm{P}_T f_0(\boldsymbol{x}_i(0)) \tag{3.24b}$$

$$\mathbb{E}[\mathrm{U}_T g(\boldsymbol{X}) \mid \boldsymbol{u}] \approx V(0) \frac{1}{N} \sum_{i=1}^{N} \mathrm{U}_T g(\boldsymbol{x}_i(0)) \, f_0(\boldsymbol{x}_i(0)) \tag{3.24c}$$

where N is the number of samples/discretizations and $V(t)$ is the volume (area) of the integration domain at time t (the support of f_t). Leveraging, again, the linearity of the SMD system, this quantity is known analytically. Given the asymptotic stability of the system, $V(t)$ decreases with time causing an increase in the PDF value of each particle, i.e. $\Psi(\boldsymbol{x}) < 0$ as discussed in Section 2.1.1. In the $T = 10$s time-frame, this does not become an

issue. However, given sufficient time, the individual values would exceed numerical limits. Figure 3.3 shows the average relative error, taken over all the possible t_{start} values, for each of the three approaches. As expected, the Frobenius-Perron and Koopman approaches evaluate to the same values and show faster convergence than the Monte Carlo simulations.

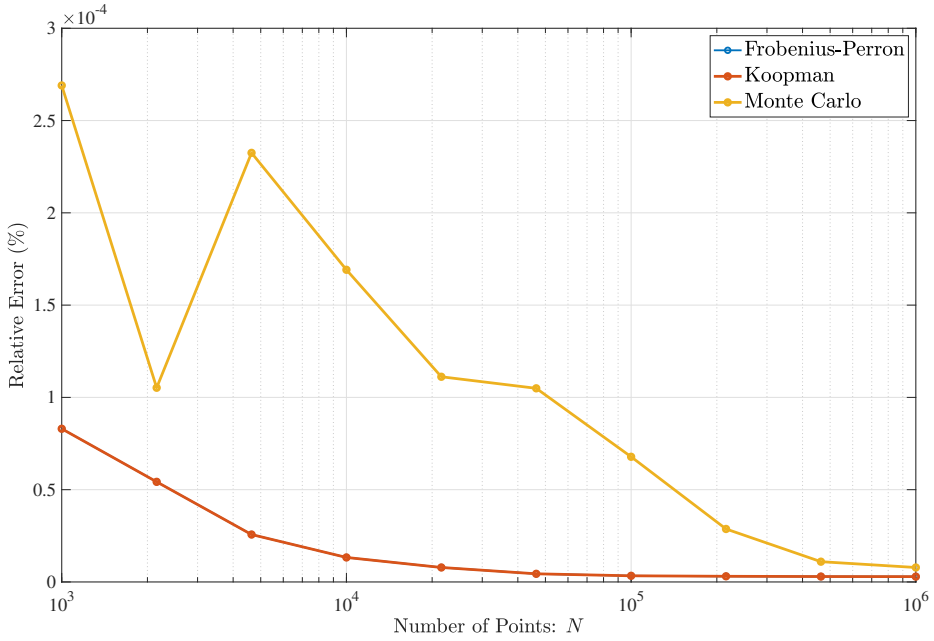


Figure 3.3: Comparison of the Frobenius-Perron, Koopman, and Monte Carlo simulation approaches evaluated numerically given N points

3.5.2 Parametric Uncertainty

This example considers the identical problem as in the previous section, except that uncertainty in the spring constant, k, is considered in addition to the initial condition, x and $\dot{x}$, uncertainty. Following the methodology of Section 2.1.4, the state-space is augmented such that $\boldsymbol{x} = [x,\ \dot{x},\ k]^\intercal$. The joint PDF quantifying the initial condition–state *and* parameter–is taken to be

$$\begin{aligned} f_0(\boldsymbol{x}) &= f_x(x) f_{\dot{x}}(\dot{x}) f_k(k) \\ &= \mathcal{N}(0.25,\ 0.0625)\mathcal{N}(0.25,\ 0.0625)\mathcal{U}(0.5,\ 1.5) \end{aligned} \tag{3.25}$$

where $\mathcal{N}(\mu,\, \sigma)$ is the Normal distribution and $\mathcal{U}(a,\, b)$ the Uniform distribution. The Gaussian distributions are truncated at $\pm 4\sigma$ to enforce a compact support over the entire augmented state domain. This results in a three-dimensional initial state-space domain that is rectangular in $\mathbb{R}^3$.

With the additional uncertainty in the parameter k, the system becomes nonlinear in the augmented state. This nonlinearity warps the initial domain such that the state space volume at the final time $T = 10$ sec is a highly complex shape. Figure 3.4 shows the discretized state-space at the initial and final times, respectively, with their corresponding joint PDF values, pushed-forward using the Frobenius-Perron operator. The complexity of this shape prohibits an analytical solution to the expected value integral since the bounds of integration are coupled to the dynamics and therefore the initial conditions, e.g. parameter k. The three numerical approaches used previously, Eq. (3.24), are employed instead.

First, the Monte Carlo simulation approximation of Eq. (3.24a) can be applied, yielding the expected value estimates shown in Fig. 3.5 for $N = 1000$. Applying the Koopman operator approach in Eq. (3.24c) is quite straightforward, since the initial volume $V(0)$ is rectangular and known exactly. The analytical trajectory solution of Eq. (3.18) is used to obtain the cost function associated with each discretized point in the initial uncertainty space (again with $N = 1000$), and the expected value is computed and shown in Fig. 3.5. Finally, the Frobenius-Perron approach using Eq. (3.24b) is more involved–in this example–due to the difficulty in computing the volume of the joint PDF support at the final time, $V(T)$. While there are methods to approximate the domain volume in the three dimensional case, using either a convex hull [83] or alpha shape function [84], these approximations may be inaccurate even with large numbers of points and they do not easily extend beyond three dimensions. Furthermore, when using the Frobenius-Perron approach, the final PDF values at $T = 10$ sec are in the range $[0, 800]$ compared to the range $[0, 40]$ at the initial time; demonstrating the growth of the PDF values as the state space contracts. For example, by increasing the damping coefficient to $c = 1.4\ Ns/m$, the final PDF values at $T = 10$ sec

were in the range $[0, 10^7]$. Using alpha shape functions to best approximate the final domain volume, the expected value estimates using the Frobenius-Perron approach are shown in Fig. 3.5, where it is evident that the distortion of the state space and growth of the PDF values lead to reduced accuracy compared to the Koopman and Monte Carlo approaches. Note that significantly more points are required using the FP operator approach to converge to the results generated by the Koopman operator or Monte Carlo simulation. Overall, these results highlight the benefit of the Koopman operator approach in avoiding problems with complex domain volumes and loss of numerical precision as the PDF evolves over time.

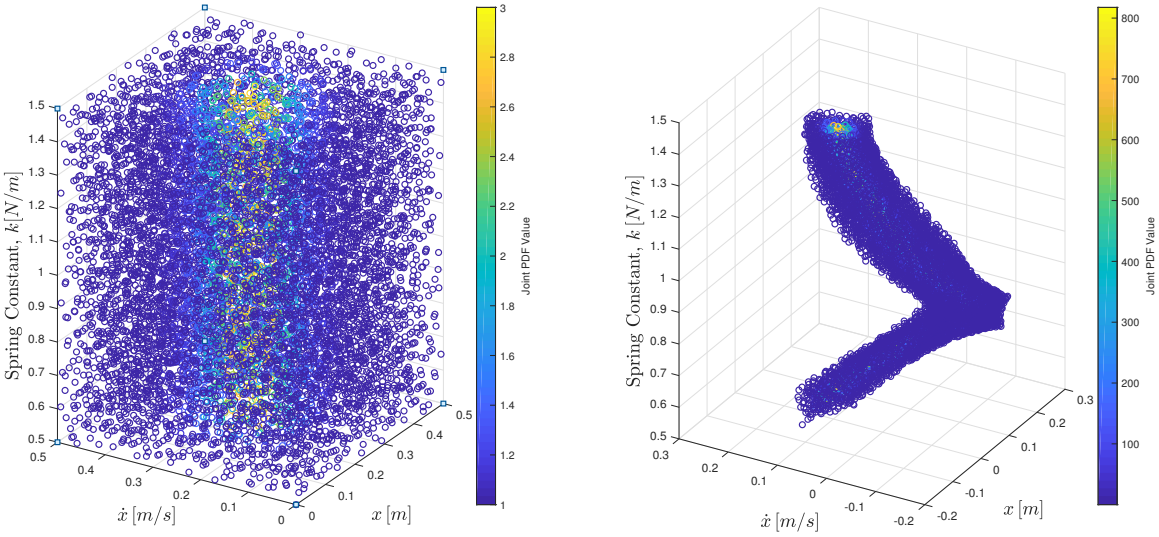


Figure 3.4: (left) Initial joint PDF with rectangular support, $N = 10{,}000$ particles realized using Halton sequence. (right) Final joint PDF at $T = 10s$, given the nonlinear mapping particles fill non-rectangular volume.

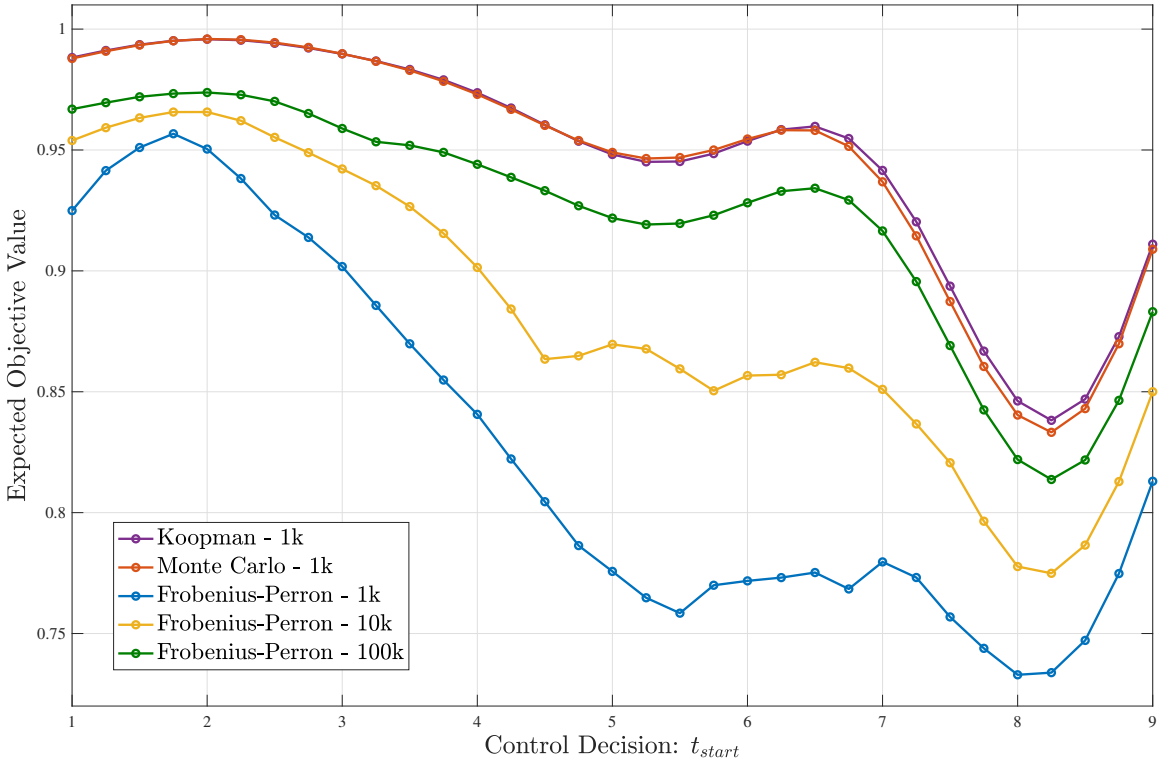


Figure 3.5: Expected values for possible control decisions (t_{start}) for parametric and initial condition uncertainty. Various N amounts for Frobenius-Perron approach highlights impact on convergence.

CHAPTER 4
APPLICATION TO PRECISION AIRDROP

A primary contribution of this work is the development of a probabilistically optimal method to the selection of the CARP, aircraft run-in, and transition altitude for ballistic airdrop. As such, a mathematical representation of the parachute-payload dynamical system must be developed. Additionally, the sources of uncertainty in the mission planning process must be identified. These sources are separated into uncertainty during release and uncertainty during descent as described below. The uncertainty during release is conditioned on the CARP location and run-in heading control decisions.

Using the parachute-payload dynamical system and associated sources of uncertainty, probabilistic planning procedures are developed for the optimal selection of the CARP location and run-in heading considering a predetermined transition altitude and an optimized transition altitude. A discussion on the numerical implementations of the algorithms is included.

4.1 Parachute-Payload Dynamic Model

The planning process developed in this work addresses high altitude, low opening (HALO) ballistic airdrop missions. However, the methodology can be easily applied to more standard airdrop missions in which the main parachute is deployed immediately after release from the aircraft. A typical HALO airdrop descent can be divided into the three stages depicted in Figure 4.1. As the package exits the aircraft, the drogue parachute inflates and the initial momentum from aircraft release dissipates, at which point the package is said to be stabilized. From here, the system descends under the drogue parachute in a quasi-steady state to the so-called transition altitude, where the drogue parachute is released and the main parachute begins to inflate. After the main parachute is fully inflated it descends

in a quasi-steady state until impact with the terrain. In this work, two different scenarios are considered. In the first scenario, the transition altitude is predetermined and constant across all bundles. In the second, the transition altitude is optimized and therefore may vary between multiple bundles. An investigation into the benefits of the optimized transition altitude scenario can be found in [20, 19, 22].

For the purposes of this paper, a number of simplifying assumptions are made. First, stabilization of the drogue-parachute system is assumed to be deterministic and happens immediately after exiting the aircraft. Second, the main parachute inflation follows a simple parametrized model as described in [22]. Lastly, the drag of the payload is neglected and the drag of the system is comprised entirely by that of the parachute. It is important to note that these assumptions pertain to the model choice and are not a reflection of the methodology itself. For example, in the case of the first assumption, a random forward-throw descent stage may be included in the model without any modification to the methodology.

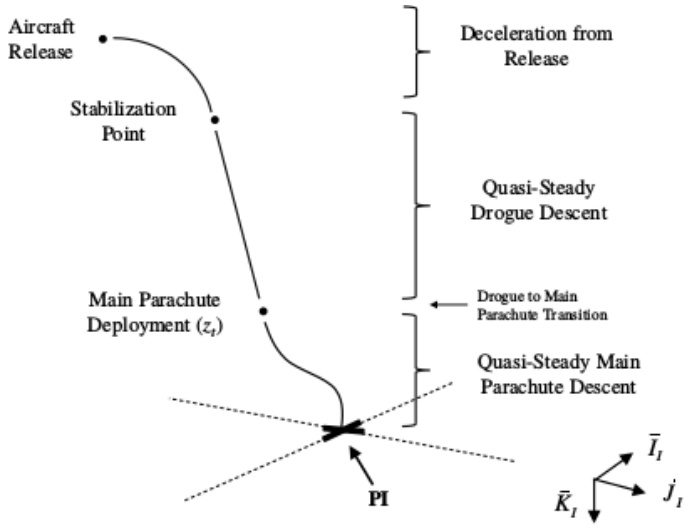


Figure 4.1: HALO airdrop stages.

Considering an inertial, non-rotating North-East-Down (NED) coordinate system shown in Figure 4.1, the equations of motion for a point-mass model representation of the parachute-

payload system are given by [22]:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \frac{-\rho C_d S \|\boldsymbol{V}_{rel}\| - 8 k_a \rho \pi R^2 \dot{R}}{2\left(m + m_a\right)} \begin{bmatrix} \dot{x} - w_x \\ \dot{y} - w_y \\ \dot{z} - w_z \end{bmatrix} + \frac{m}{m + m_a} \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.1}$$

where ρ is the air density, S is the parachute's reference area, C_d is the coefficient of drag, m is the mass of the system, m_a is the mass of the air contained in the parachute canopy (referred to as the apparent mass), R is the instantaneous parachute radius, and g is gravity. Note that the values of S and R depend on whether the drogue or main parachute is deployed at the time that the model is evaluated. A spatially-varying, three-dimensional wind field

$$\boldsymbol{w}(x, y, z) = w_x(x, y, z)\hat{I} + w_y(x, y, z)\hat{J} + w_z(x, y, z)\hat{K} \tag{4.2}$$

is considered and as such the magnitude of the airspeed vector is given by

$$\|\boldsymbol{V}_{rel}\| = \sqrt{\left(\dot{x} - w_x\right)^2 + \left(\dot{y} - w_y\right)^2 + \left(\dot{z} - w_z\right)^2} \tag{4.3}$$

The second term of the numerator in Eq. (4.1) accounts for parachute inflation. The parameter k_a is the apparent mass coefficient and is parachute-specific. During drogue and main parachute descent, this second term is zero due to $\dot{R} = 0$, while during inflation this term is non-zero. A full derivation of the inflation dynamics is provided in [22], where the change in the parachute's radius, $\dot{R}$, is defined as a function of time using an empirical model developed by Ludtke [85].

Considering a fully inflated drogue or main parachute, i.e. $\dot{R} = 0$, the quasi-steady-state package descent rate is given by [86] as

$$\dot{z}_{ss} = \sqrt{\frac{2mg}{\rho\left(z\right) C_d S}} \tag{4.4}$$

Given the model defined in Eq. (4.1), the state vector of the parachute-payload system

during quasi-steady state descent is defined as

$$\boldsymbol{s} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^\intercal \tag{4.5a}$$

$$s.t \qquad \dot{\boldsymbol{s}} = F(\boldsymbol{s}). \tag{4.5b}$$

where it is assumed that the parachute is fully inflated ($\dot{R} = 0$) and therefore R is omitted from the state vector in Eq. (4.5a). During inflation, when $\dot{R} > 0$, the instantaneous parachute radius R is explicitly included, i.e.

$$\boldsymbol{s}_R = [x, y, z, \dot{x}, \dot{y}, \dot{z}, R]^\intercal \tag{4.6a}$$

$$s.t \qquad \dot{\boldsymbol{s}}_R = \left[F(\boldsymbol{s}),\ \dot{R}\right]^\intercal. \tag{4.6b}$$

4.2 Sources of Uncertainty

The model defined above depends on many environmental and system-specific parameters. In traditional planning schemes, a *nominal* set of parameters is used during the planning process and any deviation from this set in the real, physical system will result in an outcome that differs from that predicted by the planner. However, by capturing the uncertainty with a joint probability density, the probabilistic planner can model a range of outcomes and condition the optimal solution parameters on the most likely (or expected) outcome given the current uncertainty. This subsection details the sources of uncertainty analyzed in this study, which were selected as the most influential sources of uncertainty typically present in airdrop scenarios [87].

4.2.1 Uncertainty During Descent

In the analysis by Petry [88] a comprehensive list of factors affecting airdrop accuracy is compiled and investigated. During descent the most influential factors affecting airdrop accuracy include wind, parachute performance, and Total Time of Fall, which is interre-

lated with the first two factors. Within the scope of this paper, these sources of uncertainty are captured by perturbations to the wind field forecast and uncertainty in the main parachute coefficient of drag. While these are the most influential sources of uncertainty during package descent, the proposed probabilistic framework can easily account for additional sources of uncertainty if the associated distributions (and correlations with other uncertain variables) can be appropriately quantified.

Main parachute performance and Total Time of Fall are parameterized by specifying PDFs for the main parachute's coefficient of drag, which when coupled with a wind field, affects dispersion through drift errors. In addition, the difference between the true wind field at drop time and the forecast wind field is captured by introducing perturbations to the wind magnitude and direction. By rewriting the xy-components of the wind vector from Eq. (4.2) in polar coordinates, parametric uncertainty is modeled by

$$w_m(x, y, z) = \hat{w}_m \sqrt{w_x^2 + w_y^2} \tag{4.7a}$$

$$w_\psi(x, y, z) = \arctan\left(\frac{w_y}{w_x}\right) + \hat{w}_\psi \tag{4.7b}$$

where the multiplicative wind magnitude perturbation $\hat{w}_m$ and the additive wind direction perturbation $\hat{w}_\psi$ are random variables, but are assumed constant over all altitude. This reduction of wind uncertainty to two random parameters valid over all altitudes is a simplification of the actual uncertainty in winds that exists in real-world scenarios, but is in line with those used by other mission planning tools. It should also be noted that for the remainder of this paper, the z component of the wind vector, w_z, is assumed to be zero.

Including the uncertainty in drag coefficient, the uncertain parameter vector for the airdrop problem may be defined as:

$$\boldsymbol{p} = \left[C_d,\, \hat{w}_m,\, \hat{w}_\psi\right]^\intercal \tag{4.8}$$

Assuming that the uncertainties in each dimension are independent and uncorrelated at the

drop altitude, the joint PDF quantifying the parameter uncertainty is

$$f_p(\boldsymbol{p}) = f_{C_d}(C_d)\, f_{\hat{w}_m}(\hat{w}_m)\, f_{\hat{w}_\psi}(\hat{w}_\psi) \tag{4.9}$$

While f_{C_d}, $f_{\hat{w}_m}$, and $f_{\hat{w}_\psi}$ may be any valid PDF (derived, for instance, from simulated or experimental data), in this work they are taken to be Gaussian: $f_{C_d} = \mathcal{N}(\mu_{C_d}, \sigma_{C_d})$, $f_{\hat{w}_m} = \mathcal{N}(1.0, \sigma_{\hat{w}_m})$, and $f_{\hat{w}_\psi} = \mathcal{N}(0.0, \sigma_{\hat{w}_\psi})$. Specifically, the distribution modeling the uncertainty in the coefficient of drag, f_{C_d}, is developed from experimental data [19].

Following the form of Eq. (2.22), the state-vector from Eq. (4.5a) is extended to include the parameter-vector $\boldsymbol{p}$,

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{p} \end{bmatrix} = \left[x,\, y,\, z,\, \dot{x},\, \dot{y},\, \dot{z},\, C_d,\, \hat{w}_m,\, \hat{w}_\psi\right]^\intercal \tag{4.10}$$

$$\boldsymbol{x} = \begin{bmatrix} F(\boldsymbol{x}) \\ \boldsymbol{0} \end{bmatrix} \tag{4.11}$$

where the parameters have zero dynamics (i.e., are assumed constant with respect to time). The state-vector for the inflation dynamics, Eqs.(4.6a) and (4.6b), may be extended in the same way.

4.2.2 Bundle Exit Uncertainty

In addition to the uncertainty found in the descent process, Petry [88] also identified sources of uncertainty in the package release process which must be accounted for. The bundle exit PDF formulated here explicitly describes the uncertainty in the packages' *actual* horizontal coordinates at release from the aircraft. This uncertainty is conditioned on the coordinates of the CARP, which is the *desired* package release location. This is commonly known in the airdrop community as the "CARP miss", and may be caused by pilot error in tracking the CARP location, a delay of the loadmaster releasing the packages when the CARP is crossed, time delays caused by friction as the package rolls out of the aircraft, the position

of the load in the aircraft, and other factors.

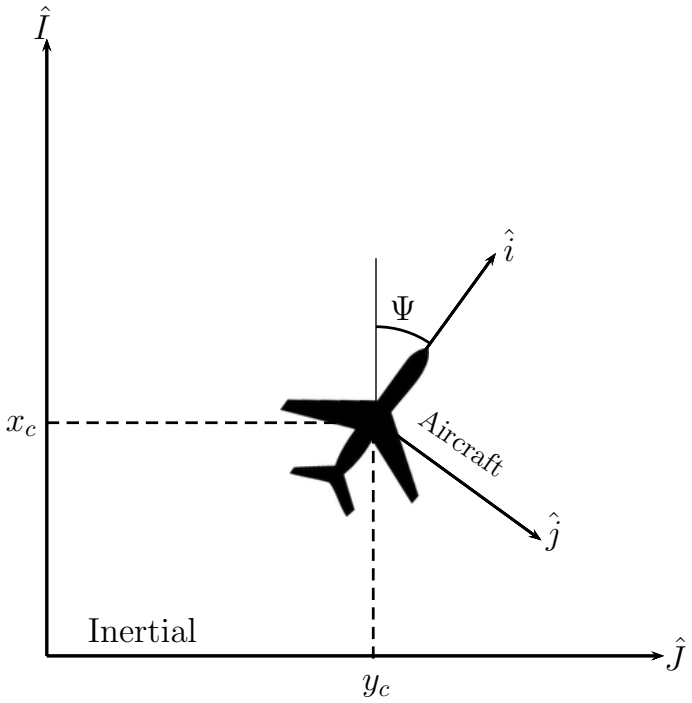


Figure 4.2: Aircraft reference fame, rotated Ψ off of the Inertial NED frame

To determine the bundle exit PDF, an aircraft reference frame is defined with its origin at the CARP location, (x_C, y_C), and its $\hat{i}$ axis aligned with the aircraft run-in, as shown in Figure 4.2. The bundle exit uncertainty is captured by the longitudinal CARP miss in the $\hat{i}$ direction, $\hat{X}_C$, the lateral CARP miss in the $\hat{j}$ direction, $\hat{Y}_C$, and the bundle exit time T_e, where the bundle exit time is defined as the difference in time between two consecutive bundles exiting the aircraft. This time is primarily driven by the pitch angle of the aircraft and the rolling friction of the cargo floor. In a multi-bundle drop, this time is considered to be constant across the stick[1]. These three random variables are taken to be Gaussian and characterized by the PDFs $\hat{X}_C \sim \mathcal{N}(\mu_{\hat{x}}, \sigma_{\hat{x}})$, $Y_C \sim \mathcal{N}(\mu_{\hat{y}}, \sigma_{\hat{y}})$, and $T_e \sim \mathcal{N}(\mu_{T_e}, \sigma_{T_e})$, respectively. These distributions are developed from experimental data [19].

In the longitudinal $\hat{i}$-direction, the PDF describing the release location of the k^{th} bundle in the stick is a function of the random variables $\hat{X}_C$ and T_e and is defined as a Gaussian

[1]The term for the collection of bundles

according to,

$$f_{\hat{x}}^{(k)} = \mathcal{N}\left(\mu_{\hat{x}}^{k},\ \left(\sigma_{\hat{x}}^{k}\right)^{2}\right) \tag{4.12}$$

where the mean and standard deviation is given by,

$$\mu_{\hat{x}}^{(k)} = \mu_{\hat{x}_C} + (k-1)\, v_g \mu_{T_e} \tag{4.13}$$

$$\sigma_{\hat{x}}^{(k)} = \sqrt{\sigma_{\hat{x}_C}^{2} + \left((k-1) v_g \sigma_{T_e}\right)^{2}} \tag{4.14}$$

and v_g is the aircraft's ground speed. Figure 4.3 shows the bundle exit PDF for a stick of eight packages and demonstrates that each package in the stick has a unique PDF describing its initial release location.

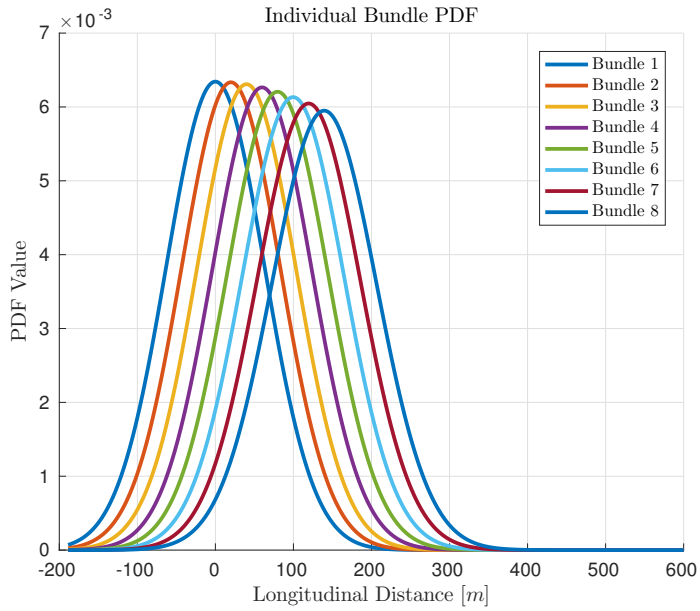


Figure 4.3: Longitudinal component of bundle exit PDF for an 8-bundle stick

Assuming that the lateral CARP miss is independent of, and uncorrelated with, the longitudinal CARP miss and package number within the stick, a joint PDF describing the uncertainty in the release location of package k in the aircraft frame may be written as $f_{\hat{x}\hat{y}} = \mathcal{N}\left(\hat{\boldsymbol{\mu}}^{(k)},\ \widehat{\Sigma}\right)$ where $\hat{\boldsymbol{\mu}}^{(k)} = \left[\mu_{\hat{x}}^{(k)},\ \mu_{\hat{y}_C}\right]^{\mathsf{T}}$ and $\widehat{\Sigma}$ is a diagonal matrix of elements

$(\sigma_{\hat{x}}^{(k)})^2$ and $(\sigma_{\hat{y}_C})^2$. For planning purposes, it is convenient to rotate this 2D Gaussian joint PDF into the inertial frame and express its mean relative to the origin of the inertial frame as follows:

$$R(\Psi) = \left[\begin{smallmatrix} \cos\Psi & -\sin\Psi \\ \sin\Psi & \cos\Psi \end{smallmatrix}\right] \tag{4.15}$$

$$f_{xy}^{(k)} = \mathcal{N}\left(R(\Psi)\,\hat{\boldsymbol{\mu}}^{(k)} + \left[\begin{smallmatrix} x_C \\ y_C \end{smallmatrix}\right],\; R(\Psi)\,\widehat{\Sigma}\,R(\Psi)^{\intercal}\right) \tag{4.16}$$

where $(x_C,\, y_C)$ is the CARP location in the inertial frame and Ψ is the run-in heading.

The joint PDF describing the total initial condition uncertainty in the augmented state space is constructed using Eqs. (4.9) and (4.16). Assuming that the parametric uncertainty in Eq. (4.9) is independent of the xy-uncertainty in Eq. (4.16), the joint PDF of the k^{th} bundle state at the drop altitude may be written as

$$f_0^{(k)}(\boldsymbol{x} \mid \boldsymbol{u}) = f_{xy}^{(k)}(x,\, y \mid \boldsymbol{u})\, f_{C_d}(C_d)\, f_{\hat{w}_m}(\hat{w}_m)\, f_{\hat{w}_\psi}(\hat{w}_\psi)\, \mathbf{1}_A(z,\, \dot{x},\, \dot{y},\, \dot{z} \mid \boldsymbol{x}) \tag{4.17}$$

where $\boldsymbol{u} = [x_C,\, y_C,\, \Psi]^{\intercal}$ and the remaining augmented state space dimensions $(z,\, \dot{x},\, \dot{y},\, \dot{z})$ are included using a characteristic function [30]. The characteristic function of the subset A, $\mathbf{1}_A$, is defined as

$$\mathbf{1}_A(\boldsymbol{x}) = \begin{cases} 1 & \boldsymbol{x} \in A \\ 0 & \boldsymbol{x} \notin A \end{cases} \tag{4.18}$$

with the subset A in Eq. (4.17) being

$$A = \left\{ z = z_d,\, \dot{x} = w_m \cos(w_\psi),\, \dot{y} = w_m \sin(w_\psi),\, \dot{z} = \dot{z}_{ss} \colon \boldsymbol{x} \in \mathcal{D} \right\} \tag{4.19}$$

where z_d is the drop altitude, $w_m = w_m(x,\, y)$ and $w_\psi = w_\psi(x,\, y)$ are from Eq. (4.7), and $\dot{z}_{ss}$ is the quasi-steady-state package descent rate under the drogue parachute defined in Eq. (4.4). It can be seen that the characteristic function provides a method that allows augmented states to be initially deterministic or a deterministic function of probabilistic states.

Using Lebesque integration, the characteristic function $\mathbf{1}_A$ ensures that $\int_\Omega f_0(\boldsymbol{x})\mu(d\boldsymbol{x}) = 1$ is satisfied.

The initial joint PDF in Eq. (4.17) is conditioned on a CARP location and run-in heading and is therefore the entry point of the control variable $\boldsymbol{u} = [x_C,\, y_C,\, \Psi]^\intercal$ into the precision airdrop problem. The goal of the probabilistic planning methodology is therefore to choose the optimal value of these three parameters given the uncertainty distributions during package release and package descent with respect to an objective function defined on the ground.

4.3 Probabilistic Planning Procedure

When formulating the probabilistic mission planning algorithm, the separation of the decision and objective spaces introduced in Section 3.1 is an important step and distinction. The measure for success in this problem is typically defined at ground–or terrain–level, while the decisions to be made are defined at the drop altitude. These spaces are connected through the dynamics of the parachute-payload system defined in Section 4.1. The uncertainty inherent in the dynamics of the system combined with the uncertainty in the actual initial conditions that result from a selected CARP and heading is the motivation behind formulating the planning algorithm probabilistically. In this section, the probabilistic planning algorithms will be discussed in detail. This includes the definition of an objective function, use of the Koopman operator to transport the function to the decision-space, and finally the steps taken to optimally determine the CARP and aircraft heading.

4.3.1 Optimization Elements

To facilitate discussion of the CARP and run-in optimization problem, the objective and decision spaces must be defined. The decision-space is taken to be

$$\mathcal{D} = \{z(0) = z_d \colon \boldsymbol{x} \in \Omega\,,\; t = 0\} \tag{4.20}$$

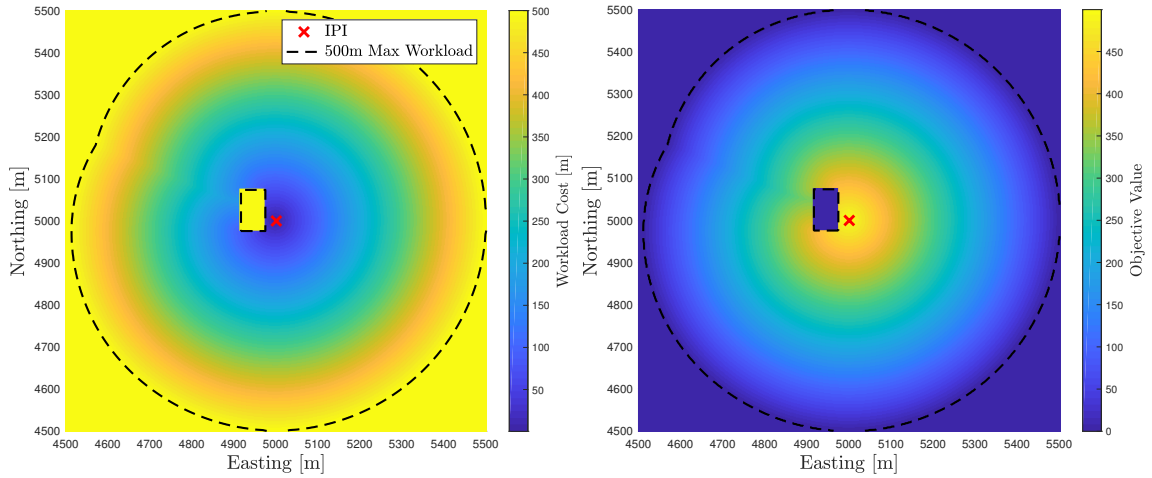
where z is a member of $\boldsymbol{x}$, Eq. (4.10), and z_d is the pre-determined drop altitude (usually selected through operational considerations). Given that mission success is usually defined as a function of where the packages land on the ground, it is natural to define the objective-space as the set of all points in the state space which have a zero altitude above the local terrain:

$$\mathcal{O} = \{z(t) - h\left(x(t),\, y(t)\right) = 0 \colon \boldsymbol{x} \in \Omega,\, t > 0\} \tag{4.21}$$

where x and y are also members of $\boldsymbol{x}$ and h is a terrain elevation function. For flat terrain, $h(x,\, y) = 0$. It is assumed that $h(x,\, y) < z_d \;\forall\, x,\, y$ and therefore $\mathcal{D} \cap \mathcal{O} = \emptyset$. The semi-dynamical system generated by the parachute-payload system's EOMs in the augmented state space (Eq. (4.5b)) connect the decision and objective spaces. The specific mapping is indexed via T, the time needed to impact terrain, $\mathrm{S}_T \colon \mathcal{D} \to \mathcal{O}$. The joint PDF quantifying the uncertainty in the parachute-payload system's augmented state, supported initially on the decision-space, is fully described in Eq. (4.17).

The objective function is a user-supplied function that allows tactical considerations to be embedded in the airdrop problem. Defined over the objective-space, $\mathcal{O}$, it is a function of x and y only for this work. However, a functional dependence on any other element of the augmented state-space Ω may be included with minimal modification. While the objective can represent any physical or non-physical quantity, one possible approach is to define a cost function based on the *workload* required to retrieve a package, where workload is a weighted function of the lateral and/or vertical distance that must be traveled from a starting location to a retrieval point [89, 82]. However, when considering a cost, the corresponding function is typically not compactly supported, a restriction discussed in Section 2.1.2. Using Eq. (2.16), the workload cost function may be inverted into a compactly supported objective function, where the maximum workload is user-specified. After the expected value of the objective function is taken, the result may be reverted back to a cost.

Figure 4.4 shows an example of inverting a cost function to an objective function. Fig-



(a) Cost function (b) Objective function

Figure 4.4: Workload cost function and corresponding objective function generated using 500m upper bound

ure 4.4a is generated by the workload cost function

$$g_{\text{WL}}(x,\, y) = D(x,\, y \mid x_{\text{IPI}},\, y_{\text{IPI}}) \tag{4.22}$$

where $(x_{\text{IPI}}, y_{\text{IPI}}) = (5000\text{m},\ 5000\text{m})$ and D is a function quantifying the travel distance from the IPI to an arbitrary $(x,\, y)$ point in meters. In the simplest of scenarios this is the Euclidean distance from the IPI. However, when obstacles are present, a method for computing the minimal travel path is needed. This can be implemented by using Dijkstra's algorithm [90] or the fast marching method [91], for example. It can be seen that there exists an untraversable region to the west of the IPI, creating an asymmetry in the objective function. Figure 4.4b shows the cost function converted to an objective function using Eq. (2.16a) with a maximum bound of $\Lambda = 500$ m. Note that as distance from the IPI increases, the function decreases to a minimum of zero satisfying the need for compact support. As highlighted in the next section and in References [89, 82], this same methodology can be used to create complex and multi-faceted objective functions capturing various combinations of tactical objectives and operational constraints.

Using the methodology in Section 3.3, inequality constraints may be optionally added

to facilitate the embedding of additional complex mission requirements into the planning procedure. These constraints may, for example, enforce specified no-fly zones or attempt to eliminate terrain impacts in specified regions. A no-fly zone would be an example of a constraint function supported on the decision-space, while eliminating terrain impacts in specified regions is an example of a constraint function supported on the objective-space and is shown in Section 5.1.3. As discussed in Section 3.3, constraints in between the decision and objective spaces may also be considered, e.g. enforcing that packages do not fly *over* civilian areas at low altitudes.

With all of the needed elements defined, the optimal xy-CARP location and run-in heading may be found by solving

$$[x_C^*,\, y_C^*,\, \Psi^*]^\intercal = \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{O}}\left[g\left(\boldsymbol{X}\right) \mid \boldsymbol{u}\right] \tag{4.23a}$$

$$= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{D}}\left[\mathrm{U_S} g\left(\boldsymbol{X}\right) \mid \boldsymbol{u}\right] \tag{4.23b}$$

where Eq. (4.23a) is the Frobenius-Perron form and Eq. (4.23b) is the Koopman form. The control variable $\boldsymbol{u} = [x_C,\, y_C,\, \Psi]^\intercal$ is found explicitly in Eq. (4.17), and is therefore considered an optimal PDF selection problem. If a spatially-varying wind field (Eq. (4.2)) is employed, in which $(x_C,\ y_C)$ will influence the specific wind velocity values the package experiences during its descent, then the problem may be considered a combined optimal system and PDF selection problem.

4.3.2 Formulating the Expected Value

With the expected value integrals of Eqs. (4.23a) and (4.23b) shown to be equivalent in Section 3.2, computational processes which implement either version of the expected value should theoretically yield the same result. However, several computational advantages can be obtained by using the Koopman operator form. This section will detail a proposed computational process to evaluate the expected values in Eqs. (4.23a) and (4.23b), and

will likewise describe the computational advantages obtained when using the Koopman operator as opposed to the Frobenius-Perron operator.

As a first step in computing the expected value, the mapping $\mathrm{S}_T \colon \mathcal{D} \to \mathcal{O}$, where T is the terrain impact time, is formed using an ensemble of trajectories. It should be noted that, from Section 3.4, the use of S_T is a slight abuse of notation given that the time to terrain impact T may be a function of the initial conditions and therefore is unable to parametrize the semidynamical system and corresponding semigroups. If this is the case, the change of variables procedure presented in Section 3.4 should be applied to resolve the conflict, e.g. reparametrizing to z from t in the case of flat terrain. Emphasizing again a key point from Section 3.4, the reparametrization of the Koopman operator semigroup is trivial. However, for notational simplicity, the system and semigroups are left parameterized by t.

For the precision airdrop problem, it is of interest to ensure that this ensemble of trajectories covers important xy-areas of the objective-space at the time of impact. To compute the trajectories, a set of N realizations $\{\boldsymbol{\omega}^{(i)}(T_i)\}_{i=1}^{N}$ is created, where T_i is the time at which the realization impacts the ground. Each realization $\boldsymbol{\omega}^{(i)}$ contains $\left[x,\, y,\, C_d,\, \bar{w}_m,\, \bar{w}_\psi\right]^\mathsf{T}$ and is an element of the subspace

$$A_T = \left\{ \begin{array}{cc} (x,\, y) \in A_{xy} & x,\, y \in \mathbb{R} \\ {C_d}^{\min} \leq C_d \leq {C_d}^{\max} & C_d \in \mathbb{R} \\ \bar{w}_m^{\min} \leq \bar{w}_m \leq \bar{w}_m^{\max} & \bar{w}_m \in \mathbb{R} \\ \bar{w}_\psi^{\min} \leq \bar{w}_\psi \leq \bar{w}_\psi^{\max} & \bar{w}_\psi \in \mathbb{R} \end{array} \right\} \tag{4.24}$$

where A_{xy} is the area of interest on the ground. The area of interest and minimum/maximum values for each parameter are scenario-specific and defined by the user. For the work in this paper, a multidimensional Halton sequence [80] is used to form the realizations. While the Halton sequence possesses good space-filling characteristics, uniformity of the discretized points is not necessary and additional points may be strategically added to increase resolution in certain areas. It is important to note that the subspace A_T is being *discretized*, rather than *sampled* from a joint PDF. As such, it is not required that the spatial density of the

realizations $\{\boldsymbol{\omega}\}$ be correlated to the joint PDF or objective function in any way.

Given that the realizations have defined terrain impact locations, trajectories from the drop altitude to ground are generated by solving a two-point boundary value problem (BVP) for each $\boldsymbol{\omega}^{(i)}$. In this work, a single shooting method with an adaptive step-size is used [92]. From Eq. (4.19), the compact support of the initial condition joint PDF is a deterministic function of $\boldsymbol{\omega}$ and can be used to populate a full augmented-state for each $\boldsymbol{\omega}^{(i)}$ at the drop altitude. The EOMs in (4.1) are forward-integrated from the drop altitude (decision-space) to terrain impact (objective-space) for each realization, creating a set of N unique trajectories through the augmented state-space.

Using (2.11) and (2.15), the ODE forms of the Frobenius-Perron and Koopman operators derived using the MOC, the forward-propagated joint PDF and pulled-back score function may be embedded into each realization. Equation (2.11) states that the joint PDF of the augmented state generated by the realization $\boldsymbol{\omega}^{(i)}(T_i)$ at ground is

$$f_{\mathcal{O}}(\boldsymbol{\omega}^{(i)}(T_i) \mid \boldsymbol{u}) = f_{\mathcal{D}}(\boldsymbol{\omega}^{(i)}(t_0) \mid \boldsymbol{u})\, e^{-\int_{t_0}^{T_i} \Psi(\tau)\, d\tau} \tag{4.25}$$

where t_0 is consistent for all realizations. Likewise, Eq. (2.15) states that realization $\boldsymbol{\omega}^{(i)}$'s objective function value at the drop altitude is

$$g_{\mathcal{D}}\left(\boldsymbol{\omega}^{(i)}(t_0)\right) = g_{\mathcal{O}}\left(\boldsymbol{\omega}^{(i)}(T_i)\right) \tag{4.26}$$

where $g_{\mathcal{O}}$ is the user-defined objective function.

The use of the Koopman operator is a natural choice over the Frobenius-Perron operator for the evaluation of the expected value of the objective function for a number of reasons. If the Frobenius-Perron form of the expected value is used, Eq. (4.23a), then the joint PDF would need to be propagated forward for each possible control decision. This is because the density evolved through the state space is a function of the control variables. As a result, the density propagation step along each realization in (4.25) will need to be

performed repeatedly, each time a new control selection is evaluated in the search for an optimal decision. Alternatively, since Eq. (4.26) does not explicitly contain the control decision $\boldsymbol{u}$, the pull-back of the score function must only be performed once. Evaluating alternative control values (CARP locations and headings) then only involves re-computing the expected value integral in Eq. (4.23b) with the specific joint density under consideration. Compared with the Frobenius-Perron form of the expected value, which requires both a density propagation step and computation of the expected value integral for each candidate control decision, use of the Koopman operator involves only re-computation of the integral and thus saves significant computational effort.

A second benefit of the Koopman operator is that the joint PDF is never explicitly propagated. During descent, the parachute system approaches steady-state values in the velocity dimensions, causing the volume of the support of the original joint PDF to decay to zero. As such, the PDF values increase drastically to preserve the unit volume constraint of the PDF. This may result in joint PDF values that are too large to work with numerically. Hoogendoorn et al. experienced a similar problem in [41]. This is an issue which will affect any system exhibiting $\Phi(\boldsymbol{x}(t)) < 0$, which includes stable systems, given sufficient propagation time. Use of the Koopman operator form of the expected value, Eq. (4.23b), avoids this issue with growth in the PDF values since the joint PDF is never propagated (i.e., Eq. (4.25) is never used in the solution procedure).

Finally, as pointed out at the beginning of this Section, a reparameterization of the system and semigroups is needed even for the simplest case of flat terrain. In the case of complex, non-flat terrain or when considering the transition altitude to be a non-constant system parameter the reparameterization becomes increasingly difficult for the Frobenius-Perron operator as the derivative of the function relating t to the new independent variable v, $dv = \dot{\phi}(t, \boldsymbol{x})dt$, is used explicitly in the MOC solution (Eq. (3.15a)). However, when using the Koopman operator, the MOC solution (Eq. (3.15b)) *does not change* when re-parameterizing. This allows for t to be used to develop the solution before *representing* it

in terms of v.

Using the Koopman operator form of the optimization problem, the expected value of Eq. (4.23b) for the airdrop planning problem is:

$$\mathbb{E}_{\mathcal{D}}\left[G\left(\boldsymbol{X}\right) \mid \boldsymbol{u}\right] = \int_{\Omega} G(\boldsymbol{x})\, f_{xy}(x, y \mid \boldsymbol{u})\, f_{C_d}(C_d)\, f_{\hat{w}_m}(\hat{w}_m)\, f_{\hat{x}_\psi}(\hat{w}_\psi)\, d\boldsymbol{x} \tag{4.27}$$

where the k has been dropped from $f_{xy}^{(k)}$ for notational simplicity and $G = \mathrm{U}_S g$ is the pulled back objective function. Since the CARP and run-in decision $\boldsymbol{u}$ only appears in the initial xy-uncertainty f_{xy}, which is known in closed form, it is beneficial to partition the full integral and create a second function dependent only on x and y,

$$g^*(\boldsymbol{x}) = G(x, y, C_d, \hat{w}_m, \hat{w}_\Psi) f_{C_d}(C_d)\, f_{\hat{w}_m}(\hat{w}_m)\, f_{\hat{x}_\Psi}(\hat{w}_\Psi) \tag{4.28}$$

$$g^*_{xy}(x,\, y) = \int_{A_{C_d}} \int_{A_{\hat{w}_m}} \int_{A_{\hat{w}_\Psi}} g^*(x, y, C_d, \hat{w}_m, \hat{w}_\Psi)\, dC_d\, d\hat{w}_m\, d\hat{w}_\Psi \tag{4.29}$$

where g^*_{xy} is the expectation of the objective function under perfect knowledge of the drop location. This is typically referred to as a conditional expectation. Substituting Eq. (4.29) into Eq. (4.27), the function may be expressed as

$$\mathbb{E}_{\mathcal{D}}\left[G\left(\boldsymbol{X}\right) \mid \boldsymbol{u}\right] = \int_{A_x} \int_{A_y} f_{xy}(x,\, y \mid \boldsymbol{u}) g^*_{xy}(x,\, y)\, dx\, dy \tag{4.30}$$

which highlights the influence of the control decision $\boldsymbol{u}$ on the expected value. Given that the function f_{xy} is known analytically, the difficulty in evaluating Eq. (4.30) comes from computing the integrals found in Eq. (4.29).

While the decision to form the realization set at ground level ensures sufficient coverage of the objective function, after solving the BVPs the realizations at the drop altitude (i.e., the origins of each trajectory in the decision-space) show no consistent structure and can be considered scattered. It is only at these N locations that the function g^* is known and therefore the integrals of g^* in Eq. (4.29) must be calculated numerically. For the purposes of

this work, the kernel-based methods presented in Section 2.2.2 are used to approximate g^* from the scattered data. Lobachevsky splines were chosen over other scattered-data approximation methods because they can be integrated analytically; this procedure is presented in detail in Section 2.2.4. Additionally, they have compact support, which leads to a sparse system of equations during the approximation process, lending itself to a tractable numerical implementation. The implementation is handle by InterpCL, the GPU-accelerated C++ library highlighted in Section 2.2.5.

Following the procedure set forth in Section 2.2.2, a kernel-based interpolant of g^* is formed from the scattered data

$$g^*(\boldsymbol{x}) \approx \sum_{i=1}^{N} c_i K(\boldsymbol{x},\, \boldsymbol{x}_i) = \sum_{i=1}^{N} c_i \prod_{j=1}^{5} \phi\left(x_j - x_j^{(i)}\right) \tag{4.31}$$

where ϕ is the Lobachevsky spline function. Since multivariate Lobachevsky spline basis functions are the product of univariate functions, single dimensions may be easily integrated. When computing the conditional expectation in Eq. (4.29), three of the five dimensions (drag, wind magnitude perturbation, and wind direction perturbation) are integrated out analytically, leaving the two-dimensional function g^*_{xy}. The spline approximation of this conditional expectation may be written as,

$$g^*_{xy}(x,\, y) \approx \sum_{i=1}^{N} \bar{c}_i\, \phi(x - x_i)\, \phi(y - y_i) \tag{4.32}$$

where the $\bar{c}_i$ coefficients are modified from the original coefficients in Eq. (4.31) according to the procedure developed in Section 2.2.4. This constitutes the *analytical* integration of the *approximated* g^* function across the chosen dimensions.

Returning to the full expected value of Eq. (4.30), the density of the uncertainty in x and y is given by the bundle exit PDF, Eq. (4.16). Substituting Eq. (4.32) into Eq. (4.30), a functional representation of the expected objective function value given the CARP and

run-in tuple $\boldsymbol{u}$ can be written as

$$E\left(\boldsymbol{u}\right) = \int_{A_x} \int_{A_y} f_{xy}(x,\, y \mid \boldsymbol{u}) \sum_{i=1}^{N} \bar{c}_i \, \phi(x - x_i) \, \phi(y - y_i) \, dx \, dy \tag{4.33}$$

In this work, the final two integrals of Eq. (4.33) are evaluated using Monte Carlo integration. Since f_{xy} is a two-dimensional Gaussian distribution, a total of N_e evaluation points are sampled uniformly within a five standard deviation ellipse centered on the mean of f_{xy}. The complete integral is thus approximated as

$$E\left(\boldsymbol{u}\right) \approx \frac{A_{5\sigma}}{N_e} \sum_{j=1}^{N_e} \left(f_{xy}(x_j, y_j \mid \boldsymbol{u}) \sum_{i=1}^{N} \bar{c}_i \, \phi(x_j - x_i) \, \phi(y_j - y_i) \right) \tag{4.34}$$

where $A_{5\sigma}$ is the area of the 5σ-ellipse, which is itself a function of the CARP and run-in tuple $\boldsymbol{u}$. The reason for this dependency is that, when assuming a constant airspeed, dropping the packages on a heading into the wind, the aircraft's ground speed is lower. Thus, the spatial uncertainty in the drop location (and thus the support of f_{xy}) caused by roll-out delays from the aircraft and other factors is less, compared to when the aircraft drops the packages with a tailwind.

4.3.3 CARP and Heading Optimization

The methodology presented to this point formulates the optimization problem and enables the planning algorithm to compute the expected value associated with a given choice of CARP and run-in heading. To determine the optimal solution for these inputs given the assumed uncertainty, the problem in Eq. (4.23b) must now be solved. Potentially, the optimal solution may be found through a gradient-based optimization routine as described in [75]. One downside to such methods is susceptibility to local maxima, which may preclude convergence to the global optimum. A second downside is that from a mission planning perspective, optimization through the use of a solver routine does not provide any type

of situational awareness to the user, who may be interested in a visual description of the expected performance as a function of CARP and run-in selection.

To address these concerns, an approximate solution to Eq. (4.23b) is found by gridding the solution space, and selecting the optimal value from the candidate solutions in the grid. The expected values associated with these candidate solutions are then used to visualize the underlying mission tradeoffs for operator situational awareness. To begin, the control domain is discretized into $N_c \times N_h$ CARP and run-in tuples, denoted by the set $\{\boldsymbol{u}_i\}_{i=1}^{N_c \times N_h}$, where N_c is the number of (x, y) CARP locations and N_h is the number of run-in headings between $[-180^\circ, 180^\circ)$. Evaluating Eq. (4.34) at each point produces a direct relationship between the potential control decisions and their respective expected objective function value.

The optimal CARP and run-in tuple may be found by querying the element in the grid that corresponds to the maximal (or minimal) value $E(\boldsymbol{u})$. Furthermore, the expected objective value (EOV) at drop altitude can be visualized by plotting $E(\boldsymbol{u})$ at each $x - y$ location for a given heading. This map provides a visualization to the user of the expected score function that results from dropping at any $x - y$ location on the map, given that heading selection and all associated uncertainty. If the user provided a cost function, which was subsequently inverted, the EOV maps may be reverted to expected cost value (ECV) maps using Eq. (2.16b). An array of these maps can be created, one for each discretized heading. Note that when using this discretization method to find the approximate optimal CARP and run-in, the error between the selected tuple and the true optimal tuple is dependent on the grid point spacing Δx_C, Δy_C, and $\Delta\Psi$ (i.e., resolution). There is a natural tradeoff between the degree of optimality of the resulting solution and the computational burden associated with solving the problem on a higher-resolution mesh.

In addition to the optimal solution and the EOV/ECV map, the so-called Launch Acceptability Region (LAR) is also of interest from an operational perspective. The LAR is a subset of the control domain in which a given set of operational constraints are satisfied.

For instance, in the case of an objective function which quantifies retrieval workload, the LAR may be defined as the set of CARP location and run-in heading tuples that result in a workload amount under a certain value. The LAR can be visualized on top of the EOV/ECV map to improve situational awareness for the user, in order to balance airdrop mission objectives with other flight path or operational constraints.

4.3.4 Transition Altitude Optimization

The CARP and run-in heading optimization procedure developed in the preceding Sections assumed a predetermined, constant transition altitude for all bundles in a stick. In [20], [19], and [82] it is shown that airdrop mission outcomes may be improved by allowing the transition altitude to be optimized, possibly varying between individual bundles in a stick. However, the developed algorithms assumed a predetermined CARP and run-in and computed the optimal transition altitude as the package descended under the drogue parachute.

Considering the transition altitude to be a control decision to be optimized alongside the CARP and run-in, such that the benefits of the transition altitude optimization are included in the planning stage, the control input vector $\boldsymbol{u}$ is augmented to include z_t, the transition altitude, i.e. $\boldsymbol{u}_z = [x_C,\, y_C,\, \Psi,\, z_t]^\intercal$. Given that the transition altitude directly affects the trajectories of the parachute-payload system, the augmented state-space must also be expanded to include z_t as a parameter.

Using the Koopman operator to pull the objective function g back to the decision-space $\mathcal{D}$, the full optimization problem to be solved may be expressed as

$$\begin{aligned}[x_C,\, y_C,\, \Psi,\, z_t]^\intercal &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \mathbb{E}_{\mathcal{D}}\left[\mathrm{U}_{\mathrm{S}(\boldsymbol{u})} g(\boldsymbol{X}) \mid \boldsymbol{u}\right] \\ &= \arg\max_{\boldsymbol{u}\in\mathcal{U}} \int_{\Omega} \mathrm{U}_{\mathrm{S}(\boldsymbol{u})} g(\boldsymbol{x}) f_0(\boldsymbol{x} \mid \boldsymbol{u}) d\boldsymbol{x} \end{aligned} \tag{4.35}$$

where $\mathrm{S}(\boldsymbol{u})\colon \mathcal{D} \mapsto \mathcal{O}$ is generated by the ODEs developed in Section 4.1. It is assumed that the transition altitude parameter is a deterministic variable–that is, no transition alti-

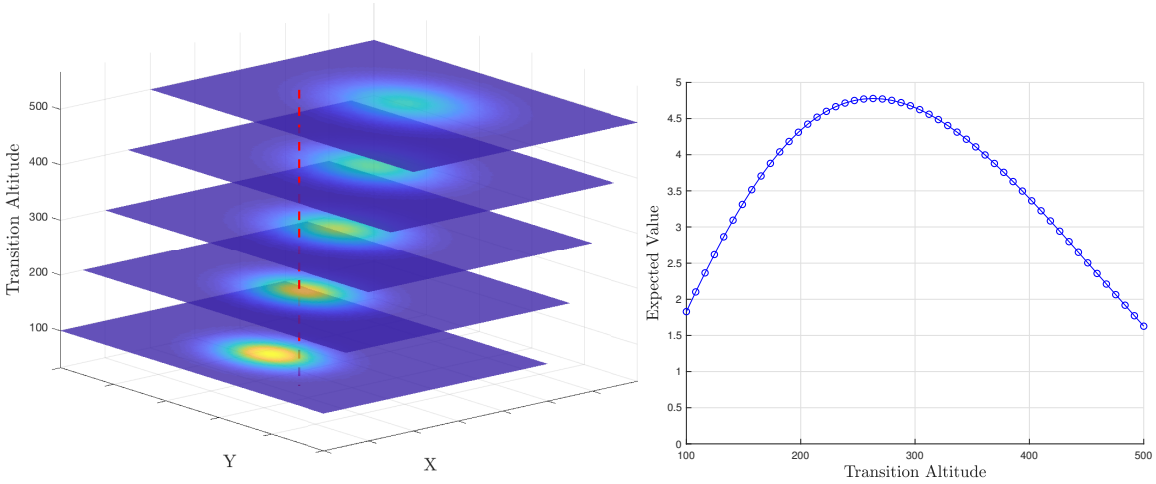
tude is *more likely* to occur than another transition altitude–and therefore its contribution to the initial condition joint PDF may be considered independent of the other variables and quantified by the characteristic function $f_{z_t}(z_t) = \mathbf{1}_A(z_t)$ where $A = [z_t^{\min},\, z_t^{\max}]$, the range of acceptable transition altitudes. In contrast to the CARP and run-in only optimization, where only the initial condition joint PDF $f_0(\boldsymbol{x} \mid \boldsymbol{u})$ was conditioned on the control input, the inclusion of the transition altitude embeds the control decision in the state-space mapping $\mathrm{S}(\boldsymbol{u})$.

To solve the optimization problem in Eq. (4.35), an approach similar to that of Section 4.3.2 may be taken. The realizations $\{\boldsymbol{\omega}_z\}$, used to numerically represent the state-space mapping S through trajectories, contain $[x,\, y,\, C_d,\, \bar{w}_m,\, \bar{w}_\psi,\, z_t]^\intercal$, where the first four elements are from the subspace defined in Eq. (4.24) and $z_t \in [z_t^{\min},\, z_t^{\max}]$. Considering that $\boldsymbol{\omega}_z \in A_Z$, where $A_Z \subset \mathbb{R}^6$ is a six-dimensional hyperrectangle, the space may be *filled* using a Halton sequence or similar discretization method.

After solving the BVPs for the set of realizations, a Lobachevsky spline interpolant must be fitted to the scattered, six-dimensional dataset in order to compute the conditional expectation of Eq. (4.29). Given the number of realizations needed to sufficiently fill the six-dimensional space, this proves to be exceptionally difficult. Through a combination of memory required to store the sparse representation of the kernel matrix and numerical precision issues when performing algebraic operations over large sets, for the purposes of this work, this procedure is considered intractable.

To circumvent these issues, the desired gridded-structure of the solution space is leveraged to breakup the single, large problem of Eq. (4.35) into a *set* of smaller subproblems. By first discretizing the range $[z_t^{\min},\, z_t^{\max}]$ into N_t candidate transition altitudes, $\{z_t^{(i)}\}_{i=1}^{N_t}$, the pull-back procedure using the Koopman operator and subsequent conditional expectation computation (developed in Section 4.3.2) is performed for each unique $z_t^{(i)}$ using the five-dimensional realizations $\{\boldsymbol{\omega}^{(i)}\}_{i=1}^{N}$. This results in a total of $N_t \times N$ BVPs to be solved. The trajectories are used to construct a set of N_t conditional expectation functions,

$\{g_{xy}^{*(i)}(x,\, y)\}_{i=1}^{N_t}$, using Lobachevsky splines on the five-dimensional datasets. Figure 4.5a shows an example of a set of conditional expectation functions. The functions have been evaluated on a set of xy-points and plotted against their corresponding transition altitudes on the vertical axis. The red, dashed line is located at $(0.475,\, 0.500)$ and serves as an evaluation point for the set of functions. Figure 4.5b plots the conditional expected value against transition altitude for the xy-evaluation point.



(a) Conditional expectation functions

(b) Evaluation curve

Figure 4.5: Conditional expectation set example. (left) The conditional expectation functions indexed by their respective transition altitudes. (right) A curve showing $N_t = 50$ conditional expectation functions evaluated at $(0.475,\, 0.500)$

Given that the transition altitude must be a scalar value, the set of conditional expectation functions must be combined into a single entity. From a point-wise $(x,\, y)$ perspective, a selection procedure must be applied to the evaluation curve shown in Fig. 4.5b that, for every xy-location, returns the *best* transition altitude and its corresponding conditional expectation. This procedure is given as

$$\left(\zeta(x,\, y),\, \bar{g}_{xy}^{*}(x,\, y)\right) = \Upsilon\left(g_{xy}^{*(1)}(x,\, y),\, \ldots,\, g_{xy}^{*(N_t)}(x,\, y)\right) \qquad \forall\, x,\, y \in \mathcal{D}_{xy} \tag{4.36}$$

where $\mathcal{D}_{xy}$ is the x and y dimensions of the decision-space. The selection criteria Υ produces two *functions*. The first function, ζ, returns the best transition altitude, from the set

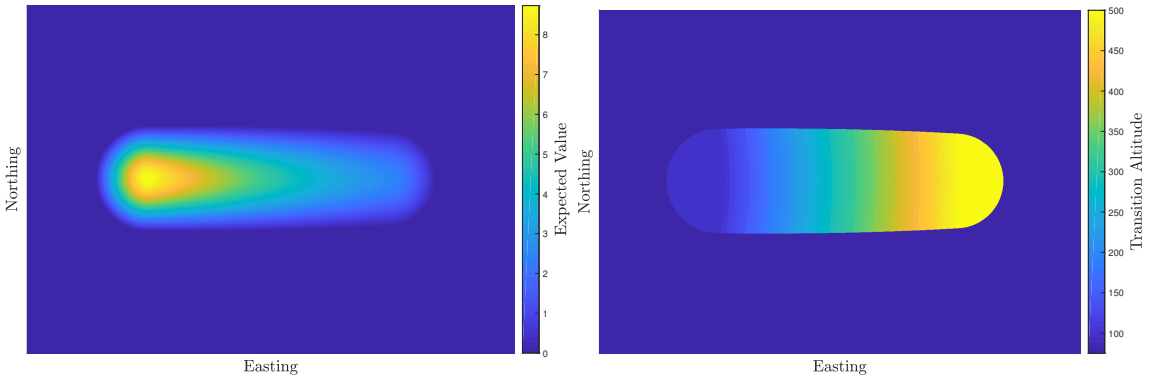
$\{z_t^{(i)}\}_{i=1}^{N_t}$, for any $(x,\, y)$ location. The second function, $\bar{g}^*_{xy}$, is the composite conditional expectation, returning conditional expectation of the best transition altitude for any $(x,\, y)$ location.

The definition of what is considered best in Eq. (4.36) is entirely dependent on the construction of Υ. For the purposes of this work, the selection criteria is taken to be, for an objective function,

$$\bar{g}^*_{xy}(x,\, y) = \max\left(g^{*(1)}_{xy}(x,\, y),\, \ldots,\, g^{*(N_t)}_{xy}(x,\, y)\right) \qquad \forall\, x,\, y \in \mathcal{D}_{xy} \tag{4.37a}$$

$$\zeta(x,\, y) = \underset{z_t \in \mathcal{Z}}{\arg\max}\left(g^{*(1)}_{xy}(x,\, y),\, \ldots,\, g^{*(N_t)}_{xy}(x,\, y)\right) \qquad \forall\, x,\, y \in \mathcal{D}_{xy} \tag{4.37b}$$

where $\mathcal{Z}$ is the set of discretized transition altitudes. For a cost function, Eqs. (4.37a) and (4.37b) would use $\min$ and $\arg\min$, respectively. Figure 4.6 shows an example of this selection process using the conditional expectation function set from Figure 4.5. Comparing Figure 4.6a to any single plot from Figure 4.5a, the operational benefit of allowing for an optimized transition altitude can be easily seen.



(a) Composite conditional expectation (b) Transition altitude schedule

Figure 4.6: Selection criteria using $\max$ and $\arg\max$. (left) The composite conditional expectation function $\bar{g}^*_{xy}(x,\, y)$. (right) Transition altitude schedule $\zeta(x,\, y)$. Dark blue region scores below a threshold, any transition altitude will obtain an equally low score.

The selection criteria defined by Eqs. (4.37a) and (4.37b) can be considered an overall ideal scenario and it assumes that the bundles will know their exact $(x,\, y)$ location exiting the aircraft. This may be handled operationally, for example, by equipping the bundles

with a GPS receiver to supply their location. However, given that the GPS data is inherently uncertain, Υ should be constructed using expected value integrals to more accurately represent the real-world scenario. This is left to future research and elaborated on in Section 6.2.4.

Using the composite conditional expectation, the functional relationship between a CARP and run-in tuple, $\boldsymbol{u}$, and the best expected objective function value may be expressed, similar to Eq. (4.33), as

$$E_z(\boldsymbol{u}) = \int_{A_x}\int_{A_y} f_{xy}(x,\, y \mid \boldsymbol{u})\bar{g}^*_{xy}(x,\, y)\, dx\, dy \tag{4.38}$$

where f_{xy} is the bundle-exit PDF, developed in Section 4.2.2. Equation (4.38) may be evaluated in a similar fashion to Eq. (4.34).

Following the procedure developed in Section 4.3.3, the remaining dimensions of the solution space, $\{x_C,\, y_C,\, \Psi\}$, are gridded to form a set candidate solutions. The total expected value is computed for each using Eq. (4.38). The optimal CARP and run-in may then be found by querying the element in the grid with the maximal (or minimal) expected value. The EOV/ECV maps described in Section 4.3.3 may be created from the candidate solution grid.

In addition to providing an optimal CARP and run-in heading that are *aware* of the ability to optimally select the transition altitude, the procedure developed above produces the optimal transition altitude schedule (i.e. $\zeta\colon (x, y) \mapsto z_t^*$). While this schedule may be used to plan the transition altitude a priori (before exiting the aircraft), by computing the expected value of ζ for each candidate solution using the bundle-exit PDF, for this work it is assumed that the optimal transition altitude is determined *as* the bundle exits the aircraft by evaluating the optimal transition altitude schedule at its drop location.

4.3.5 Extension to Multiple Bundles

The CARP location, run-in heading, and, possibly, transition altitude optimization problems have been, up to this point, with respect to a single bundle and in Section 4.3.2 the bundle position k was removed from the bundle exit PDF $f_{xy}^{(k)}$ for simplicity. By adding k back in explicitly, the functional form of the expected value integral for a given control selection becomes dependent on the bundle position,

$$E^{(k)}\left(\boldsymbol{u}\right) = \int_{A_x}\int_{A_y} f_{xy}^{(k)}(x,\, y \mid \boldsymbol{u}) g_{xy}^{*}(x,\, y)\, dx\, dy \tag{4.39a}$$

$$E_z^{(k)}\left(\boldsymbol{u}\right) = \int_{A_x}\int_{A_y} f_{xy}^{(k)}(x,\, y \mid \boldsymbol{u}) \bar{g}_{xy}^{*}(x,\, y)\, dx\, dy \tag{4.39b}$$

where Eqs. (4.39a) and (4.39b) are developed in Sections 4.3.2 and 4.3.4, respectively. Since both equations may be evaluated using Eq. (4.34) for each bundle, the z subscript will be dropped from Eq. (4.39b) for the following developments.

Considering a stick with n_b bundles, the optimal CARP location and run-in heading problem is extended to multiple bundles by

$$[x_C^*,\, y_C^*,\, \Psi^*]^{\intercal} = \underset{\boldsymbol{u}\in\mathcal{U}}{\arg\max}\, \beta\left(E^{(1)}(\boldsymbol{u}),\, \ldots,\, E^{(n_b)}(\boldsymbol{u})\right) \tag{4.40}$$

where the function β creates a composite value for the entire stick from the individual bundle expected values. The choice of β provides another input for tactical considerations to be embedded into the planning procedure. In this work, β is taken to be a weighted sum,

$$\beta\left(E^{(1)}(\boldsymbol{u}),\, \ldots,\, E^{(n_b)}(\boldsymbol{u})\right) = \sum_{k=1}^{n_b} w_k\, E^{(k)}(\boldsymbol{u}) \tag{4.41}$$

where $w_k = 1/n_b$ would produce the arithmetic mean of the expected objective function values for each bundle. The bundle weighting factors w_k provide an operator with a mechanism to weight the importance of one bundle in a stick relative to others.

Inequality constraints for multiple bundles may be handled in a similar fashion. Considering that $E_q^{(k)}(\boldsymbol{u})$ is the expected value of the constraint q for bundle k given $\boldsymbol{u}$, then an inequality constraint imposed on the stick may be written as

$$\Gamma\left(E_q^{(1)}(\boldsymbol{u}),\ \ldots,\ E_q^{(n_b)}(\boldsymbol{u})\right) \leq \lambda \tag{4.42}$$

where Γ creates a composite value for the stick. Considering the uncertain nature of the system, probability of occurrence constraints are of interest in this work. In evaluating a constraint over all n_b bundles in a stick, De Morgan's laws [45] may be used to properly combine the probabilities of constraint violations from each package. In this case, let E^k be the probability of bundle k causing a constraint violation. Then De Morgan's law asserts that the value of Γ, evaluated as

$$\Gamma\left(E_q^{(1)}(\boldsymbol{u}),\ \ldots,\ E_q^{(n_b)}(\boldsymbol{u})\right) = 1 - \prod_{k=1}^{n_b}\left(1 - E_q^{(k)}(\boldsymbol{u})\right) \tag{4.43}$$

is the probability of *any* bundle in the stick initiating the constraint violation, and should be used to evaluate whether a candidate CARP and run-in solution satisfies a probabilistic inequality constraint.

Considering the case of optimizing the transition altitude for multiple bundles, given that the optimal transition altitude schedule is developed using the set of *conditional* expectation functions, before the k-dependent bundle-exit PDF is used, it is valid for all bundles in a stick and is assumed to be used for each.

4.3.6 Implementation

The probabilistic planning procedure presented in this Section, while leveraging many benefits provided by the Koopman operator, still presents a difficult and computationally complex problem to implement. In addition to the InterpCL library, which provides a generalized framework for the kernel-based approximation methods discussed in Section 2.2.2,

a code-base containing various simulation and computation routines was developed for– and fundamental to–this work. The core functionality of the code-base is twofold. First, it provides a generalized simulation toolkit that may be used to numerically solve a system of ODEs given a large set of initial conditions. This is needed when representing the Frobenius-Perron and Koopman operators generated by the ODEs (the semidynamical system) numerically and when running Monte Carlo simulations for comparison purposes. Second, various precision airdrop specific programs and subroutines implement the planning procedures developed in the preceding Sections.

The simulation toolkit is written in C++ and implements a fourth-order Runge-Kutta (RK4) forward-integrator [93, 94] with comprehensive event detection. To make solving a large number (on the order of 10^6) of trajectories tractable, the toolkit is designed to target GPU devices using VexCL[2], a C++ vector expression template library [95, 96]. In addition to abstracting memory management away from the user, VexCL provides convenient mechanisms to build GPU-kernels (used by the forward integrator and for event detection) using just-in-time (JIT) compilation, providing a means for run-time specific improvements. The forward-intergator kernel is generated from the fixed time-step RK4 given by Odeint[3], a modern library for solving ODEs numerically [97]. This differs from the adaptive time-step scheme used by default in MATLAB's well-known `ode45` [98], choosing the tighter control provided by the former when dealing with event detection.

Events are a fundamental part of the toolkit and represent a structure with which to input and output data, e.g. initial conditions, final conditions, and observables. Used to implement the decision and objective spaces, and possibly constraint spaces, from Section 3.1, an event is said to occur when its equality $h_e(t,\, \boldsymbol{x}) = 0$ is satisfied and its derivative $\dot{h}_e(t,\, \boldsymbol{x})$

[2]https://github.com/ddemidov/vexcl
[3]http://www.odeint.com

is positive or negative as specified. Numerically, the zero-crossing is identified by

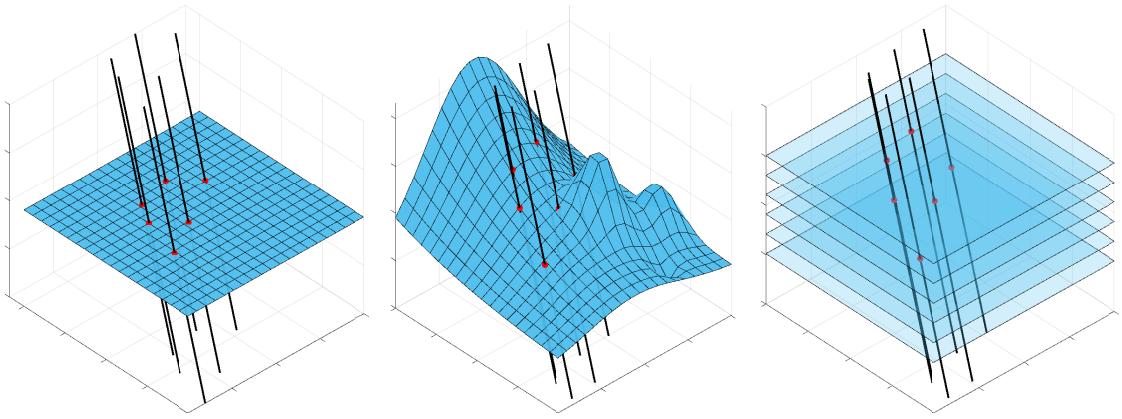
$$\frac{h_e\left(t_k,\, \boldsymbol{x}_k\right)}{h_e\left(t_{k-1},\, \boldsymbol{x}_{k-1}\right)} \leq 0 \tag{4.44}$$

$$h_e\left(t_k,\, \boldsymbol{x}_k\right)\, \Delta h \geq 0 \tag{4.45}$$

where $k \geq 1$ specifies the current time-step and Δh is 1, -1, or 0 and specifies whether the crossing in the positive, negative, or either direction, respectively, is sought. When Eqs. (4.44) and (4.45) are satisfied, the state of the system at the event is approximated by linear interpolation between $(t_{k-1},\, \boldsymbol{x}_{k-1})$ and $(t_k,\, \boldsymbol{x}_k)$ and recorded.

For the precision airdrop problem, three event types are used frequently: Constant Slice, 2D Surface, and Variable Slice. Figure 4.7 shows an example of each, respectively, with $\boldsymbol{x} \in \mathbb{R}^3$. The Constant Slice and Variable Slice events are similar considering they are both defined by $h_e(t,\, \boldsymbol{x}) = z - z_e = 0$ (where z is a state in $\boldsymbol{x}$) but differ in that z_e is constant across all particles for the Constant Slice event and may vary between particles for the Variable Slice event. Given that the event equation h_e is evaluated at every time-step for every particle, its execution must be efficient. If the function is known analytically, which it is for the Constant and Variable slice events, it may be easily made into a GPU-kernel using VexCL. For the 2D Surface event, whose event equation may be written as $h_e(t,\, \boldsymbol{x}) = z - h_z(x,\, y) = 0$, where $\boldsymbol{x} = [x,\, y,\, z]^\intercal$, the function h_z may not be known analytically, as is the case when it represents real-world terrain, such as in the form of Digital Terrain and Elevation Data (DTED) [99]. In this scenario, the terrain data is stored on the GPU as an image in what is known as *texture* memory, which allows for extremely fast bilinear interpolation. Using this memory type, the GPU-kernel performs the function evaluation via interpolation with negligible overhead.

Using the parachute-payload system, a full trajectory from the drop altitude to terrain impact is implemented in stages, similar to the spring-mass-damper system in Section 3.5. The full descent, introduced in Section 4.1, is partitioned into three stages: drogue



(a) Constant Slice (b) 2D Surface (c) Variable Slice

Figure 4.7: Simulation toolkit event examples. Black lines represent trajectories from initial conditions and red dots signify the event for each.

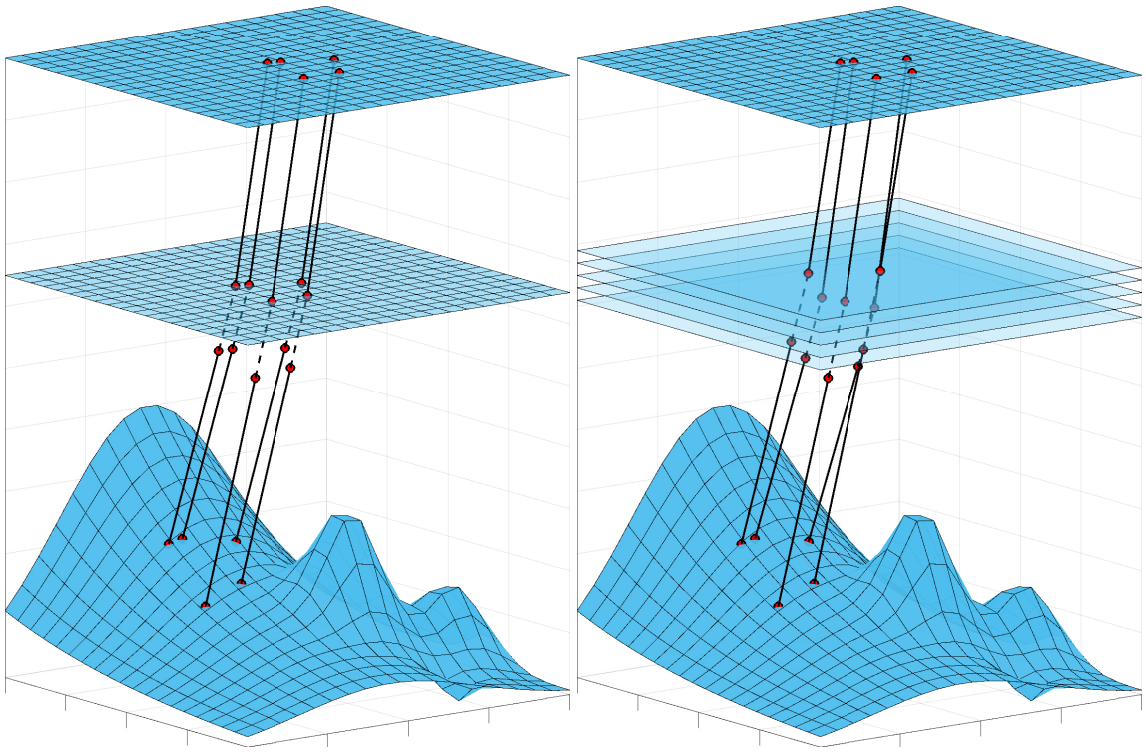
parachute descent, main parachute inflation, and main parachute descent. These stages are defined by four key events: the initial condition event at the drop altitude, the transition altitude event initiating main parachute inflation, the main parachute inflation event signaling the parachute is fully inflated, and the terrain impact event terminating the trajectory. Table 4.1 shows these stages and their corresponding initial and final events. For the drogue and main parachute descent stages, the system is defined by the ODEs in Eq. (4.5b), which do not include the instantaneous parachute radius in the state vector. During the inflation stage the parachute radius is explicitly included and the system is defined by the ODEs in Eq. (4.6b).

	Events		
Initial	Transition	Inflation	Impact
Drogue Parachute Descent			
	Main Parachute Inflation		
		Main Parachute Descent	

Table 4.1: Parachute-Payload stages defined by key events

Figure 4.8 shows examples of trajectories, events, and stages used in the optimization procedures described in the preceding sections. Figure 4.8a shows the scenario when the CARP and run-in are to be optimized with a predetermined transition altitude. As such, all particles begin their inflation stages at the same altitude. However, given differing velocities

and parameters at the start of the event, the time–or altitude change–to fully inflate will vary between the particles. Figure 4.8b shows the scenario when the transition altitude is to be optimized alongside the CARP and run-in. Discussed in Section 4.3.4, the transition altitude is included as a parameter and defines the slice event uniquely for each particle. The terminating event for both scenarios is impact with non-flat terrain.



(a) Constant Transition Altitude (b) Variable Transition Altitude

Figure 4.8: Parachute-payload model descent (solid lines) and inflation (dashed lines) stages for constant transition altitude across all particles (left) and variable transition altitude (right). The three stages are defined by four key events (red dots).

The MOC ODEs for the Frobenius-Perron and Koopman operators are solved alongside the system ODEs using an arbitrary initial condition as it is shown in [89] that the values may be rescaled later. With the trajectories of the parachute-payload system captured by its events, the objective function may be evaluated on the terrain impact event and the Koopman operator values of the other events set accordingly. This procedure, and many supporting functions, are implemented in Python as it allows for fast prototyping of

possible objective functions and the inclusion of other, advanced libraries. After the numerical application of the Koopman operator, the InterpCL library is used to perform the conditional expectation computations.

The evaluation of the full expected value using the condition expectation, Eq. (4.34), is a computationally intensive task given a large number candidate solutions (fine resolution), number of bundles, and number of Monte Carlo integration points. As such, the corresponding program is written in C++ and targets GPUs using OpenCL. The use of local and texture memory spaces are used to improve run-times. The program produces data in a per-bundle format so that it may be combined according to Section 4.3.5 and is saved in case of re-planning. Considering an 8-bundle mission with 60 candidate run-in headings, the size of the saved file is 1-2GB depending on the CARP resolution.

4.3.7 Real-Time Transition Altitude Optimization

Previous work by Leonard et al. additionally considered the problem of optimizing the transition altitude during descent under the drogue parachute and is developed in detail in [20]. Instead of pulling an objective or cost function back to the drop altitude, the algorithm pulls them back to the set of candidate transition altitudes. The conditional expectation functions are then constructed for each and transferred to a computer onboard the package that is responsible for real-time transition altitude optimization. In [20], the conditional expectation functions are called *marginal* slices.

The transition altitude optimization algorithm executed onboard the package proceeds as follows. During descent under the drogue, the current state of the bundle is measured and the package trajectory is then propagated from this measured state to an altitude of $z_t^{\min}$ under drogue parachute dynamics only. This propagation makes use of an analytical descent solution that predicts the drogue trajectory using a radial basis function approximation [19]. The benefit of this analytical trajectory propagator is that the solution time is deterministic and several orders of magnitude faster than that achieved through numerical integration,

making it practical for a real-time implementation on a low-cost embedded computer. A key assumption in the development of the analytical solution in [19] is that winds do not vary spatially in x or y. However, because the analytical model is used to propagate the drogue descent only, this assumption results in minimal errors since the package exhibits little total translation in the x and y directions under the drogue compared to its motion under the main parachute. A complete description of this analytical descent propagator, as well as an exploration of the errors introduced by the spatially-varying wind assumption, is provided in [19] and omitted here for brevity.

Evaluating each conditional expectation function at the $(x,\, y)$ location where the predicted trajectory intersects its corresponding altitude z-plane, a real-time expected value versus transition altitude relationship, similar to Figure 4.5b, generated. This curve is continually recomputed based on feedback from onboard position and velocity sensors. The work in [20] investigates two optimal transition altitude selection methods. The first is similar to the selection process described in Section 4.3.4, where the optimal transition altitude is taken to be the $\arg\max$ of the real-time expected value curve.

The second method investigates *shaping* the bundle impact dispersion into a desired distribution. In this case, the objective function on the ground is a two-dimensional PDF in x and y defining how the user would like the bundle terrain impacts to be distributed. As an example of when this may be useful, consider a humanitarian operation in which aid is to be distributed over a certain area. To accomplish this, the optimal transition altitude is taken to be a random sample *drawn* from the conditional expectation curve. When using this method, packages will probabilistically achieve the desired ground distribution (along their line of control on the ground) given a large enough number of drops.

In practice, the selection process–using either method–is performed each time a new state measurement is received and the conditional expectation curve is recomputed.The main parachute is inflated when the measured altitude, z, crosses over the current optimal transition altitude, z_t^*. This recalculation process allows the optimal transition altitude to be

updated as unexpected winds and dynamic model errors cause the actual descent trajectory to depart from the predicted trajectory.

CHAPTER 5
RESULTS

To demonstrate the capabilities of the proposed planning algorithm, simulation examples are presented. The examples consider three *real-world* scenarios. The first example is a simple scenario in which the cost function is defined by the radial distance from a desired impact location, and is intended to provide a useful comparison against standard deterministic planning schemes. Next, a scenario with non-flat terrain and a complex objective function is considered, highlighting the operational flexibility of the proposed planning methodology and its benefits over deterministic planning. Finally, a scenario is considered in which the planner minimizes an expected workload value while enforcing a probabilistic obstacle impact constraint. This type of scenario would traditionally require an operator to manually plan a mission and iteratively verify the obstacle impact probability through Monte Carlo, while in this case the probabilistic planning algorithm can solve the constrained optimization problem directly in a single planning cycle. The final scenario is considered for the CARP and run-in optimization only, and the full CARP, run-in, and transition altitude optimization is left for future work.

All cases in this section consider a HALO airdrop with a drop altitude of 10,000 ft, unless stated otherwise. For each scenario, a three-dimensional, spatially-varying wind field generated by the Weather Research and Forecasting Model [100] is used. The specific wind field used is detailed in [101]. Additionally, the random parameters (variables) during descent are taken to be Gaussian as described by the statistics in Table 5.1. The statistics for the main parachute C_d uncertainty are taken from [19]. The random variables used to construct the bundle exit PDF, needed to quantify the uncertainty in x and y at aircraft release, are also considered to be Gaussian and described by the statistics in Table 5.2. The release condition uncertainty in Table 5.2 is derived from the information provided in [19].

Table 5.1: Random variables during descent

Variable	μ	σ
Main parachute coefficient of drag (C_d)	1.0487	0.0422
Wind magnitude perturbation ($\hat{w}_m$)	1.00	0.0500
Wind direction perturbation ($\hat{w}_\psi$)	0.00 rad	0.0873 rad

Table 5.2: Random variables used in bundle exit PDF

Variable	μ	σ
CARP longitudinal miss ($\hat{X}_C$)	0.00 m	62.897 m
CARP lateral miss ($\hat{Y}_C$)	0.00 m	26.777 m
Package release time (T_e)	0.18 s	0.0300 s

An additional example demonstrating the real-time transition altitude optimization using the algorithm from Section 4.3.7 is included at the end of chapter. The scenario considers uncertainty during descent only from the random variables listed in Table 5.1. The focus of the simulation example is to shape the bundle dispersion into a desired distribution. Various additional examples of the methodology may be found in the work by Leonard et al. [20].

5.1 Solution Set: CARP & Run-in Optimization

A solution set to the three example scenarios using optimized CARP location and run-in heading with a predetermined transition altitude is presented first. A transition altitude of 1,640ft (500m), unless otherwise stated, is used for every bundle.

From a practical perspective, it is important to briefly characterize the computational performance of the above algorithm given the planning time constraints of airdrop missions. The trajectory simulation portion of the algorithm is implemented in a parallelized manner for graphics processing units (GPUs) through the OpenCL framework. The implementation used here is able to compute 2 million trajectories in approximately 200 sec on an AMD Radeon Pro 560 GPU. Using a more capable NVIDIA K40, the same 2 million trajectories can be found in approximately 71 sec. Runtime of the remainder of the planning process

is dependent on many parameters such as the spline approximation parameters, xy-CARP resolution, number of run-in discretizations, and number of bundles in a stick. In general, the time required to perform a full probabilistic planning process is approximately 1 to 2 hours on a desktop computer with a 3.5 GHz 6-core Intel Xeon processor, 64 GB of RAM, and two NVIDIA Tesla K-series GPUs.

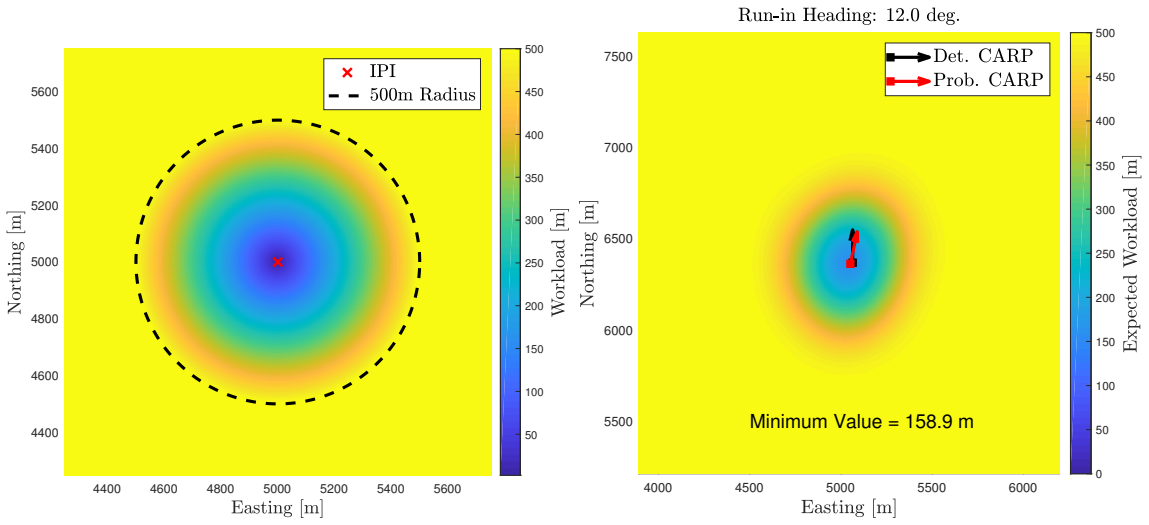
5.1.1 Radial Case

This example scenario considers a radial distance cost map where impact locations are penalized by their radial distance from the intended point of impact (IPI): $x = 5000$ m, $y = 5000$ m. Figure 5.1a shows the workload (cost) map generated by,

$$g_R(x, y) = \sqrt{(x - 5000)^2 + (y - 5000)^2} \tag{5.1}$$

where the radial distances are saturated at 500 m (as depicted by the dashed black line). The cost function is inverted according to Eq. (2.16a). The corresponding objective function is pulled-back through the parachute-payload system dynamics using $N = 512,000$ realizations. The total EOV for each control decision is computed considering a stick of $n_b = 8$ bundles and then reverted to expected workload (cost) using Eq. (2.16b).

Figure 5.1b shows the expected radial distance for every xy-CARP location for the optimal run-in of 12 deg. Two CARP locations are indicated. The first is the deterministic CARP. This is determined by solving a single BVP for the CARP location such that a package released from this location with mean system parameters lands with a score of 500 (the maximum achievable). An aircraft heading into the predominant wind is used for this calculation. The second location in Fig. 5.1b is the fully probabilistically optimal CARP. It can be seen that the xy-locations are nearly the same, which is not surprising with this scenario since in the case of relatively simple objective functions and low-complexity wind fields the mean solution will not differ substantially from that which maximizes the



(a) Cost function

(b) Expected radial workload map

Figure 5.1: (left) Radial workload function at ground (0ft). The workload is saturated at 500m. (right) Expected workload at drop altitude (10,000ft), optimal CARP and run-in are shown.

expected value.

Monte Carlo simulations were performed for each CARP (250,000 8-bundle sticks, 2M total simulations), where the uncertain descent parameters were sampled from the distributions in Table 5.1 and the initial xy-locations were sampled for each package using the bundle exit PDF, described in Section 4.2.2, and Table 5.2. Figure 5.2 shows an empirical cumulative density function (CDF) created from the mean radial distance for each stick, for both the probabilistic and deterministic Monte Carlo results. The probabilistic CARP and run-in solution results in roughly the same 50% circular error probable (CEP) as the deterministic solution, 155.4 m versus 156.1 m, respectively. These similar results, as well as the almost identical shapes of the two CDFs (Figure 5.2), show that the probabilistic planner, in this baseline scenario, will perform similar to current deterministic planning methods.

For the probabilistic solution, the planning methodology returns an expected workload of 158.9 m. Compared with the mean of the Monte Carlo results of 157.8 m, this represents a difference of 1.1 m, or a relative error of 0.69%. Considering the xy-CARP locations

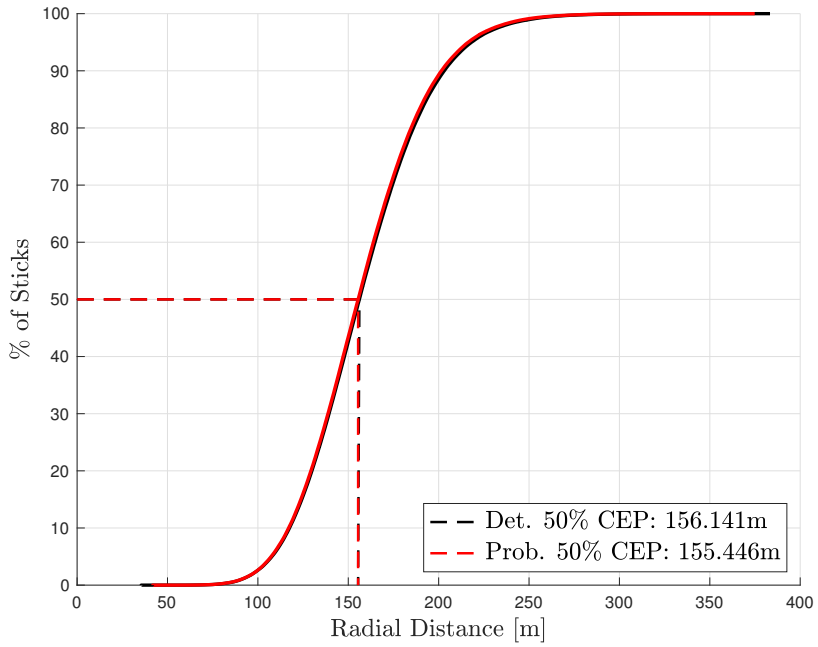
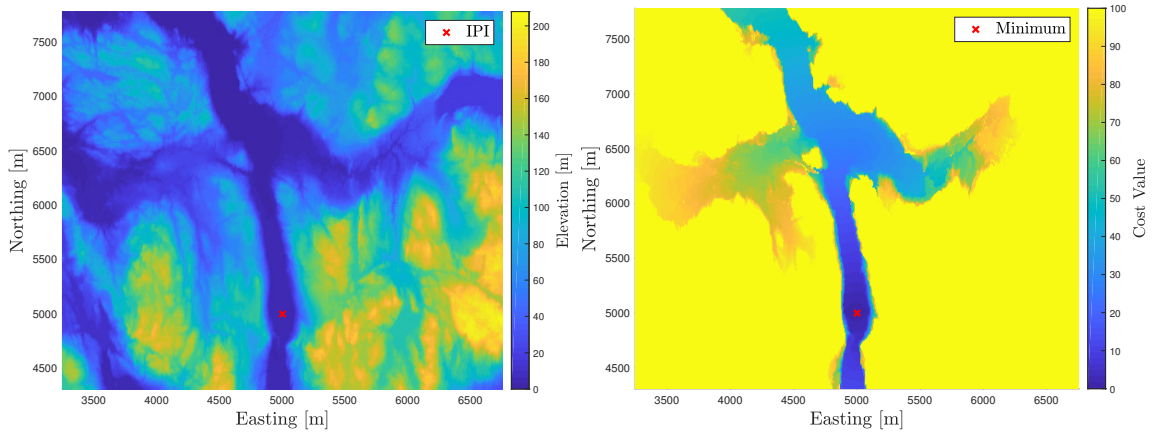


Figure 5.2: Empirical CDF of average radial workload for each stick.

were discretized on a 2 m grid, this error is within expected tolerances and provides confidence that, if the uncertainty distributions are correct, the planner can accurately predict the expected performance of the system.

5.1.2 Valley Case

To demonstrate the level of complexity that the proposed planning methodology can handle, this example considers a scenario involving non-flat terrain and a more complex cost function. Figure 5.3a shows an elevation map of a notional valley of interest. The objective function is constructed by first considering an IPI at (5000 m, 5000 m), then using Dijsktra's algorithm to compute the minimal travel cost from the IPI to every point in the set $\{x_i,\, y_i,\, z_i = h_v(x_i,\, y_i)\}_{i=0}^{nx \times ny}$, where x and y are gridded with nx and ny unique values, respectively, and h_v is the valley function depicted in Figure 5.3a. The travel costs between each point is a composite function of distance and slope, penalizing steep vertical travel heavily. The cost function is constrained and normalized such that the IPI has a cost of zero and high-elevation regions have a maximum cost of 100, shown in Figure 5.3b.



(a) Elevation data. IPI at (5000m, 5000m)

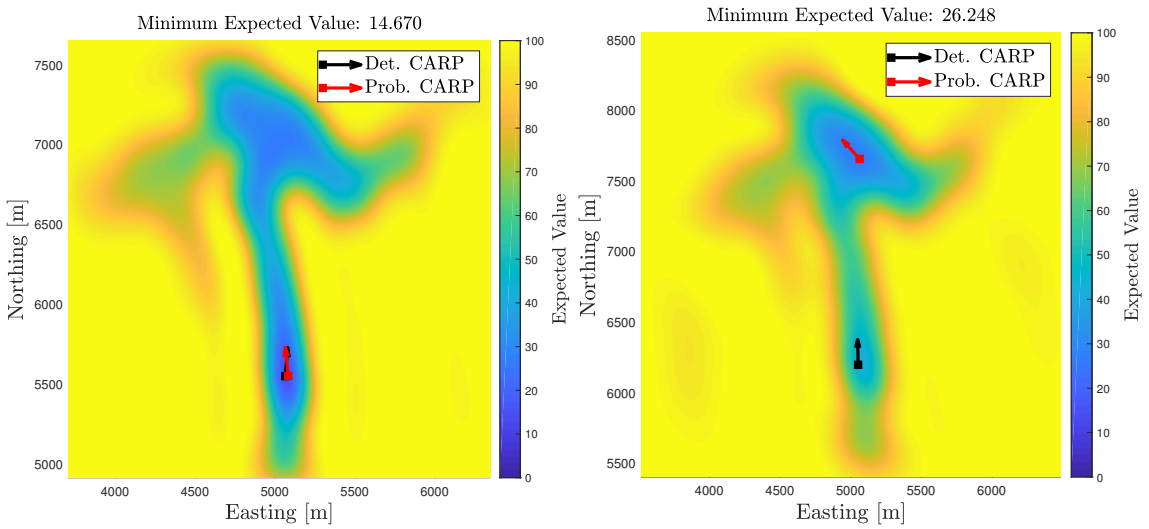
(b) Cost function. Minimum score at IPI

Figure 5.3: Complex, non-flat terrain scenario and associated workload function

Using $N = 512,000$ realizations, the cost function is pulled-back through the dynamics using a predetermined transition altitude of 400m. Note that when evaluating the cost function, the three-dimensional locations of the samples on the terrain are considered (i.e., the trajectory simulations account for impact on non-flat terrain). To demonstrate the capability of the probabilistic planner to adapt to varying levels of uncertainty, the cost function is pulled back to two different candidate drop altitudes, 3,000 ft and 10,000 ft.

Figure 5.4a shows the ECV map of the optimal run-in (-6 deg.) for a drop altitude of 3,000 ft, as well as the deterministic and probabilistic CARP solutions. It is evident once again that the deterministic and probabilistic solutions do not vary drastically. Although uncertainty is present in the drop procedure and during descent, given the relatively low drop altitude the system is not exposed to uncertain winds for a significant period of time. As a result, the expected workload value is quite low, and the induced dispersion is small enough so that the planner targets the IPI in the narrow valley. Using the probabilistic solution, the user should expect to recover an average ECV of 14.67 for an 8-bundle stick.

While low drop altitudes reduce sensitivity to uncertainty during descent, it may also pose more operational risk to the aircraft and thus higher drop altitudes are generally preferred. When dropping at higher altitudes, however, greater dispersion is induced. Figure 5.4b shows the ECV map for the optimal run-in for the 10,000 ft drop altitude case.



(a) ECV map for 3,000ft drop altitude

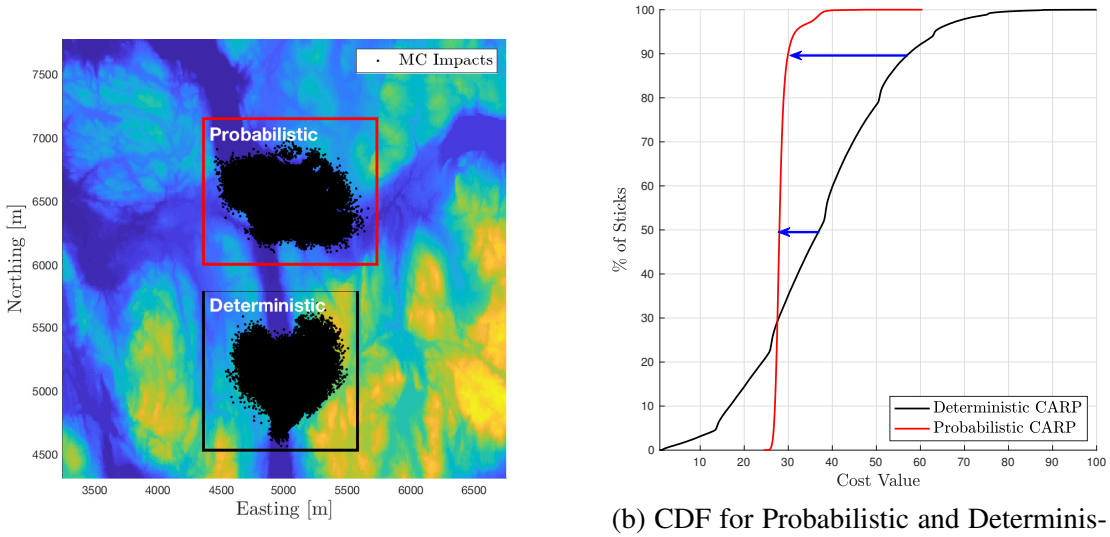
(b) ECV map for 10,000ft drop altitude

Figure 5.4: Expected workload maps of optimal headings for 3,000ft and 10,000ft drop altitudes

This figure shows that, given the increased dispersion, targeting the high-valued narrow valley is no longer the optimal solution. According to the user-supplied cost function, the possibility of landing near the desired IPI no longer outweighs the risk of landing on the mountain-tops, which incurs a large penalty. The probabilistic planning methodology automatically adapts the optimal CARP and run-in solution to target the higher-cost, but wider northern region to minimize the expected workload under the increased uncertainty without any user interaction. This stands in contrast to the deterministic planner, which continues to target the IPI since it has no mechanism to condition the CARP and run-in solution on the underlying uncertainty distributions. This ability to adapt to varying levels of uncertainty is one of the key strengths of the probabilistic planner over deterministic methods.

To compare the probabilistic and deterministic solutions for a 10,000 ft drop altitude, Monte Carlo simulations were performed. For both CARP and run-in solutions, 250,000 8-bundle sticks were simulated from the drop altitude to ground with their terrain-impact locations and workload function values recorded. Figure 5.5a shows the terrain-impacts of the two solutions, with the deterministic solution still targeting the valley and the probabilistic solution targeting the northern region. The average cost function value for every

stick is computed and used to create the empirical CDF shown in Figure 5.5b.



(a) Monte Carlo impact locations

(b) CDF for Probabilistic and Deterministic solutions

Figure 5.5: Terrain impact locations and resulting empirical CDFs for 10,000ft drop altitude

Important statistics from Figure 5.5b are listed in Table 5.3 and show the trade-offs between the two solutions. First, considering minimum values, the probabilistic solution's lowest impact value is 24.4 while 20% of the deterministic solution's impacts obtained a lower workload function value than that. More generally, the top (lowest) 10^{th} percentile of the deterministic solution's impacts is a full 10 units less than that of the probabilistic solution. However, when considering the bottom (greatest) 10^{th} percentile, the probabilistic solution outperforms the deterministic solution by over 27 units. This near-elimination of the higher-valued tails pushes the probabilistic solution's mean lower than that of the deterministic solution. In other words, the probabilistic scheme naturally trades the ability to minimize its score in a low number drops for increased robustness of the overall mission.

Table 5.3: Deterministic and Probabilistic MC impact statistics

Solution	10th Percentile	Mean	90th Percentile
Deterministic	16.73	36.99	57.32
Probabilistic	26.96	28.42	29.95
Change:	+10.23	-8.57	-27.37

The Monte Carlo simulations also provide a method to check the accuracy of the mis-

sion planner given this complex scenario. The planner computed an expected *objective* function value of 73.75 for the optimal CARP and run-in while the Monte Carlo *objective* function results show an average value of 71.58, producing a relative error of 3.0%. When comparing the cost function values, 26.25 from the planner and 28.42 from Monte Carlo, the relative error is increased to 7.6% through the reversion process.

5.1.3 Constrained Case

A final example explores the planning methodology's ability to handle probabilistic constraints. In this scenario, the packages should land so as to minimize the travel distance from an IPI while avoiding an obstacle or exclude region near the IPI. The allowable probability that any bundle in the stick lands within the obstacle will be given by a tolerance between 0 and 1. For this example, a drop altitude of 10,000 ft and transition altitude of 1,640 ft (500 m) for an 8-bundle stick is used.

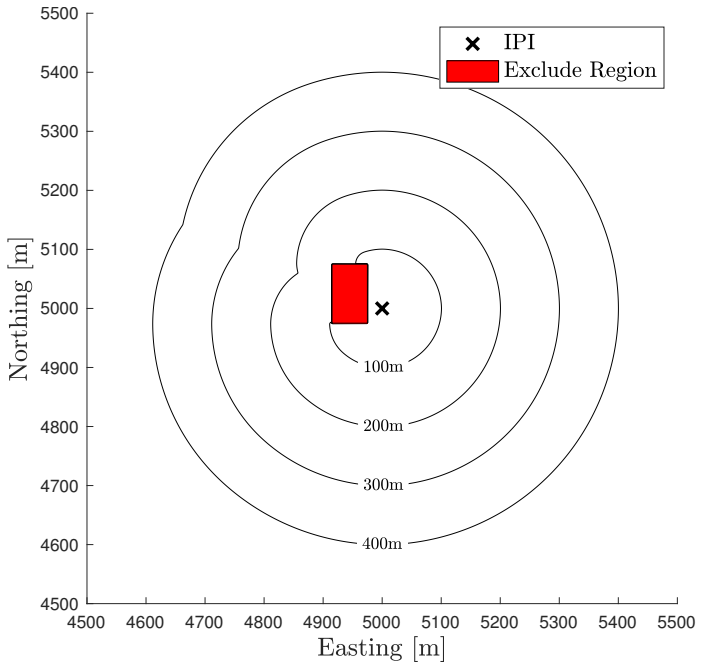
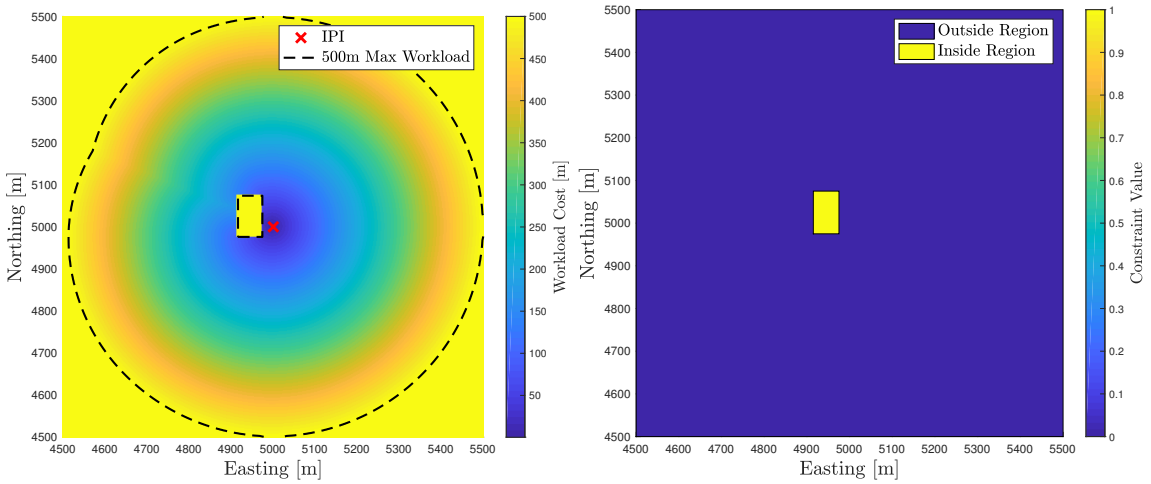


Figure 5.6: Exclude region mission scenario: width 60 m, height 100 m

Figure 5.6 shows the mission scenario. An IPI is located at (5000 m, 5000 m) and the distance traveled (workload) to retrieve the bundles is to be minimized. Additionally, the



(a) Objective function, max value 500

(b) Constraint function

Figure 5.7: Exclude region case objective and constraint functions

exclude region is a 60 m $\times$ 100 m area near the IPI that cannot be traversed and as such the probability that a package lands in the area is included as a constraint. The optimal CARP and run-in decision is formulated using an inequality constraint as defined in Eq. (4.42). The inequality constraint function is to describe the probability that any bundle in the stick lands in the exclude region. As such, DeMorgan's law (Eq. (4.43)) is used to compute the stick's composite value.

The cost function g, shown in Figure 5.7a, is constructed by computing the geodesic travel distance from the IPI, given that the exclude region cannot be traversed. It is then inverted using a maximum allowable cost of 500 m. The constraint function c, shown in Figure 5.7b, represents the probability of being within the exclude region at ground level and is therefore 1 inside the region and 0 outside the region. The user's tolerance to a bundle landing within the exclude region is quantified by λ_e.

To compute the optimal CARP and run-in in this constrained case, both the objective (inverted cost) and constraint functions must be pulled-back. The objective function is pulled-back using 512,000 realizations uniformly distributed over the uncertainty space using a 5-dimensional Halton sequence. The constraint function is pulled-back using 614,400 particles, half of which are uniformly distributed over the landing area and half of which

are distributed within a ±5 m band around the exclude region's border. As discussed in Section 4.3.2, there is no requirement for the discretized particles to be located in any specific area of the uncertainty space. Given the sharp changes of the constraint function, and the fact that the spline interpolation suffers from Gibbs phenomena [62], additional particles around the jump help to mitigate overshoot effects and result in improved accuracy of the spline fitting (and subsequent integration) process.

Figure 5.8 shows the ECV map with overlaid contour lines of the expected constraint function value map for a single heading. The contour lines depict the values of the inequality constraint function–to satisfy the probabilistic constraint for a given λ_e value, the CARP would need to be located outside the contour. Naturally, as the constraint becomes more strict the area of the drop-zone that violates the constraint increases. However, considering that the expected constraint value map has a maximum value of 43.8%, if the mission designated that the probability of a package landing in the exclude region be less than 50%, for example, the constraint never restricts the solution and can be neglected. If the constraint is ignored, the planner identifies that a minimum average retrieval distance of 195m per stick is possible with an optimal CARP selection. As the constraint is made more strict, the operator will need to accept that the average expected workload will increase as the CARP is moved farther away from the unconstrained optimal location.

To show the trade-off between a stricter exclude region impact tolerance and the expected workload, the planner is used to compute the optimal probabilistic solution for λ_e = 10%, 5%, and 1%. Figure 5.9 shows the resulting ECV maps of the optimal run-in for each tolerance value and corresponding Monte Carlo results. For each scenario, 250,000 8-bundle sticks are simulated from the 10,000 ft drop altitude to terrain impact (flat terrain, in this case). At ground, the cost function is evaluated at the impact locations. If any bundle in the stick lands in the exclude region an impact counter is incremented and the eight xy-impact locations for that stick are plotted in blue. On the left-hand-side of Fig. 5.9, the green circle denotes the optimal CARP solution if no constraint is enforced for the heading

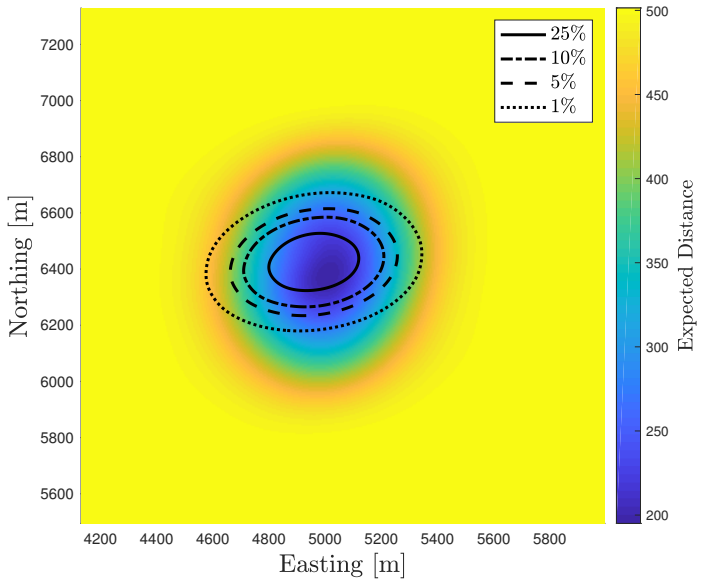


Figure 5.8: ECV map with overlaid expected constraint value contours, for a single heading

shown by the red arrow. The red point and arrow show the optimal CARP and run-in for the constrained case. Note that in the constrained cases, the CARP is located at the lowest expected value of the cost function outside of the area eliminated due to the constraint, which turns out to be on the border of the region. As the region in black grows (as λ_e is reduced by the user), the constrained solution for the CARP becomes farther from the unconstrained solution. This is corroborated by the Monte Carlo impacts on the right of Fig. 5.9, which show the dispersion pattern moving away from the IPI (and also the obstacle area) as the risk tolerance is decreased.

The Monte Carlo impact percentages and expected workloads for these cases are shown in Table 5.3. Note that the percentage of impacts inside the obstacle observed in Monte Carlo is very close to that predicted by the planner. Also note that the percent error between the expected workload predicted by the planner and that observed in Monte Carlo is quite low considering the computational process of interpolating and integrating high-dimensional, scattered data of functions with sharp jumps, as found in both the constraint and cost functions. For each tolerance value λ_e, the planning methodology was able to automatically find an optimal CARP and run-in that achieves the lowest expected work-

load while satisfying the probabilistic constraint. With all of the required data computed up-front, the probabilistic solution for any λ_e can be quickly found by querying a new minimum argument of the cost data masked by λ_e without the need to entirely re-plan or include a human-in-the-loop to verify the resulting dispersion pattern. This ability to re-plan the mission quickly as new levels of risk are established is another key strength of the proposed algorithm.

Table 5.4: Planner and Monte Carlo comparison

λ_e	% Impact Inside Constraint	Workload (*planner*, *MC*)	% Error
10%	9.90% (24752 / 250000)	220.9m, 214.0m	3.2%
5%	5.01% (12515 / 250000)	237.6m, 230.8m	2.9%
1%	1.02% (2560 / 250000)	272.2m, 266.4m	2.2%

The example cases shown in this section highlight some important features of the planning methodology introduced in this work. First, all three example cases demonstrate how mission objectives are quantified within the planner through the cost and constraint functions, and how the probabilistic solution is conditioned on these inputs. By explicitly including the objectives in the planning stage, the user is provided with greater levels of information as an output in the form of the expected workload maps. Secondly, the developed methodology possesses the ability to incorporate a wide variety of uncertainty which affects the airdrop mission planning process, from uncertainty in wind fields to uncertainty in drop location caused by variation in the time that a loadmaster releases the bundles. Finally, by computing and storing the expected value maps offline, planners may react quickly to changing scenarios and risk tolerances without the need to completely replan the mission.

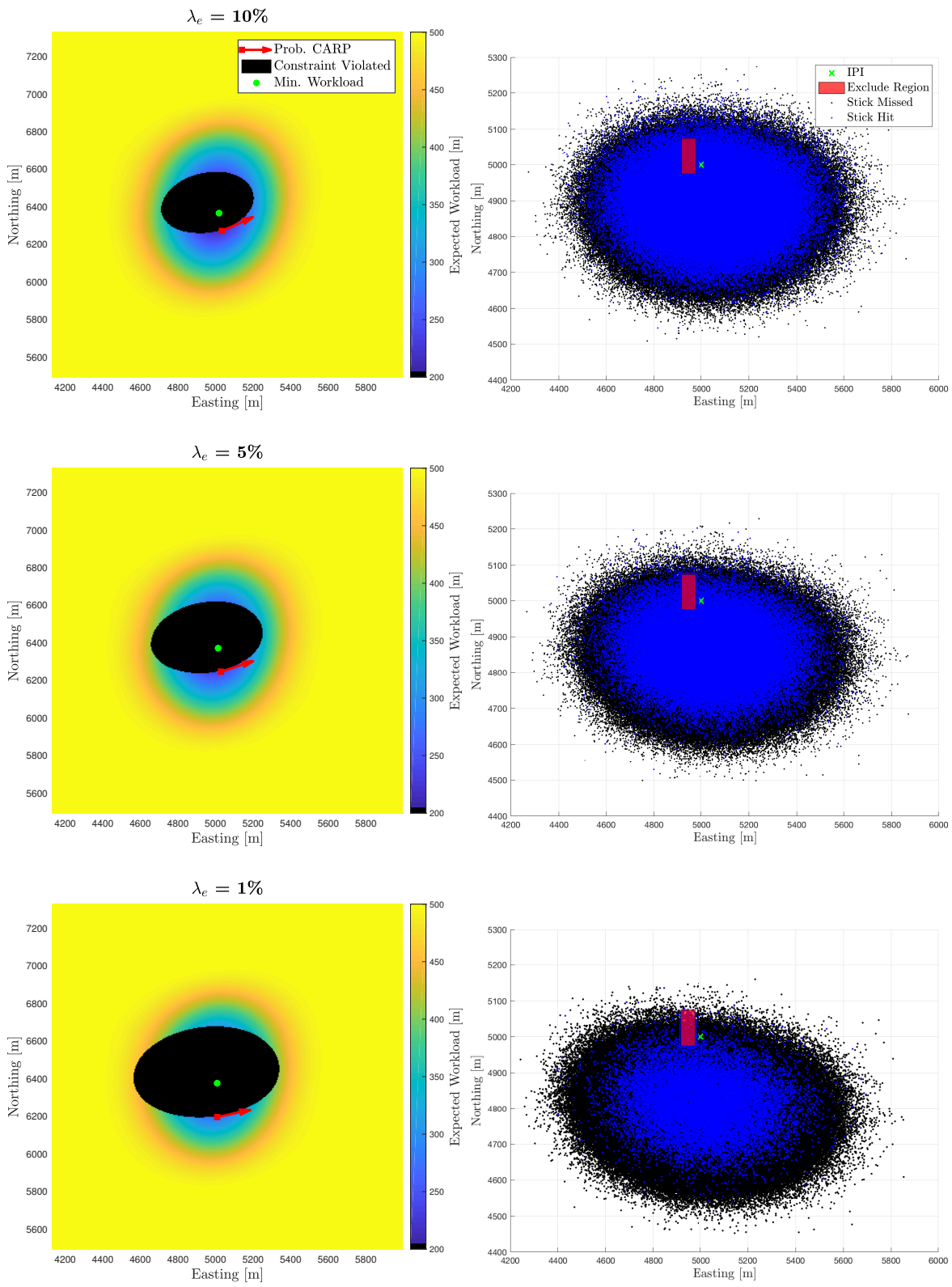


Figure 5.9: ECV map with constraint and associated Monte Carlo results for λ_e = 10%, 5%, 1%

5.2 Solution Set: CARP, Run-in, & Transition Altitude Optimization

The example scenarios presented in the preceding Section assumed a constant, predetermined transition altitude. Presented now is a solution set where the full CARP, run-in heading, and transition altitude optimization procedure is performed. A range of transition altitudes, consistent with each example scenario, are considered. These ranges are representative of real-world missions and taken from [22, 19].

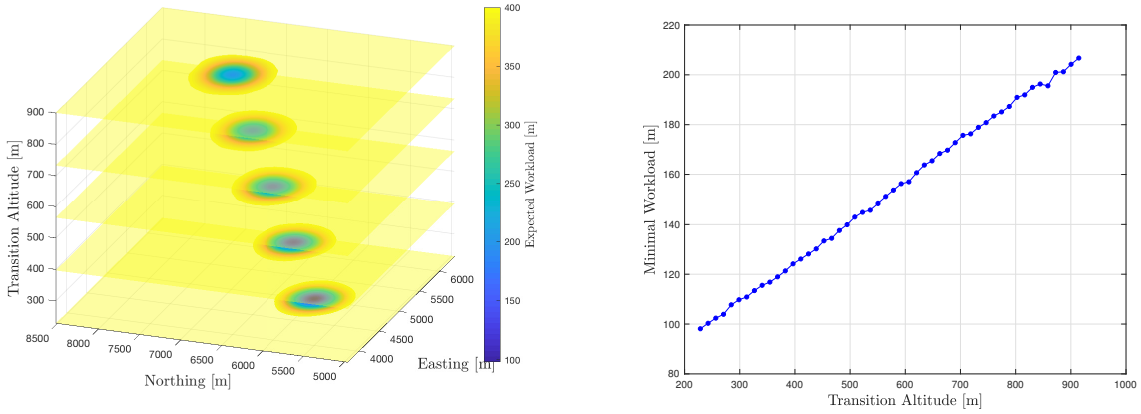
Given the added complexity of the transition altitude optimization, the probabilistic planning procedure requires more time. While dependent on a number of factors (including number of realizations used, CARP locations and headings considered, number of bundles, etc.), the largest run-time factor is the number of candidate transition altitudes. The kernel-based interpolation and integration procedure must be applied to each and may take upwards of an hour to compute. Fortunately, this process may be run in parallel, leveraging multiple devices. Using a K40 and K20 GPU in combination, the average time to compute 50 conditional expectation functions for the three example cases is roughly 12-18 hours. The remaining planning steps then take an additional 1-2 hours.

5.2.1 Radial Case

This example scenario is identical to the mission presented in Section 5.1.1. The radial distance cost function, defined in Eq. (5.1) and shown in Fig. 5.1a, is used with an IPI of $x = 5000$m, $y = 5000$m and is saturated at a workload of 400m. A total of 50 candidate transition altitudes are considered ranging from 228.6m to 914.4m (750ft to 3000ft), giving a resolution of approximately 14m (46ft). At the steady-state descent under the drogue parachute, this corresponds to an approximate 2Hz update rate and is considered sufficient.

Using $N = 350{,}000$ realizations per transition altitude, for a total of $N_t \times N = 50 \times 350{,}000 = 17{,}500{,}000$ trajectories, the radial cost function is pulled-back to a drop altitude of $10{,}000$ft using the Koopman operator. A conditional expectation function is

computed for each candidate transition altitude and aggregated into a set. Figure 5.10a shows a selection of these functions. It can be seen that as the transition altitude increases, the area of minimal workload (blue) moves into the dominant wind, blowing north to south. Figure 5.10b shows that, in addition to the translation of the minimum workload location, the minimum workload value increases with increasing transition altitudes. Given the uncertainty during descent under the main parachute, any additional time (or altitude) spent in the uncertain winds, for example, will increase dispersion and, in turn, increase the minimal expected conditional cost.



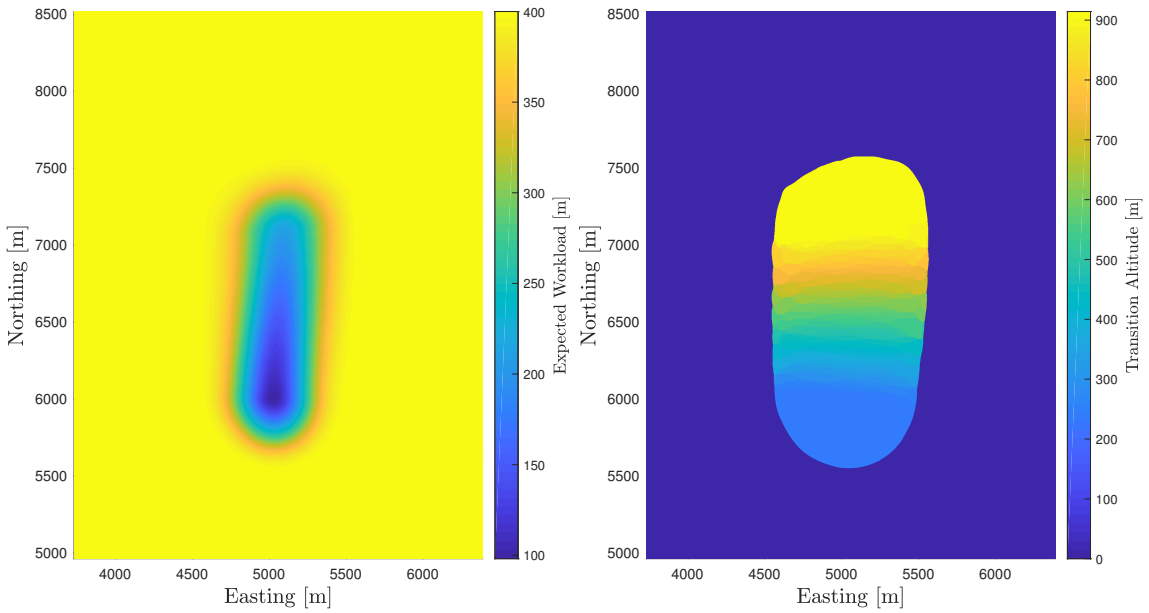
(a) Conditional expectation functions

(b) Minimum conditional workload

Figure 5.10: Radial distance cost function scenario. Dominant wind is from north to south.

A composite conditional expectation function and corresponding optimal transition altitude schedule are constructed using the $\max$ and $\arg\max$ functions, discussed in Section 4.3.4, respectively. Figure 5.11a shows the radial composite conditional expectation function. The area of minimal workload is *stretched* back, into the dominant wind, from the global minimum location, forming a shape much different than the function defined on the ground. Figure 5.11b shows the corresponding optimal transition altitude schedule. The optimal transition altitude increases linearly when moving directly north, into the dominant wind. This is due to the bundle's line of control, investigated in [20], being primarily influenced by the wind field. This is also investigated deterministically in [22] as the Wind-Drift coefficient. The dark-blue regions in Fig. 5.11b, showing a transition altitude of

0m, correspond to the saturated workload value and will score equally using any transition altitude.



(a) Composite conditional expectation

(b) Transition altitude schedule

Figure 5.11: Radial cost function scenario composite conditional expectation function and corresponding transition altitude schedule *selected* using $\max$ and $\arg\max$, respectively.

Figure 5.12 shows the ECV map for every xy-CARP location for the optimal run-in heading of 0 deg for an $n_b = 8$ bundle drop. Figure 5.13a shows the CARP and flightpath (red line), where the length of the flightpath is taken to be the 95% confidence interval of the bundle-exit PDF, overlaid on the transition altitude schedule. The optimal CARP location assumes a transition altitude of 350ft, the lowest value in the range. The transition altitude schedule is evaluated along the flightpath and shown in Figure 5.13b. The red lines mark the transition altitudes of the 8 bundles in the stick assuming nominal performance (no CARP miss or release-time variations).

To quantify the improvements over the constant, predetermined transition altitude scenario from Section 5.1.1 Monte Carlo simulations were run from the optimized CARP, run-in heading, and transition altitude schedule. Considering the added level of complexity, $1{,}000{,}000$ 8-bundle sticks (8M simulations total) were drawn from the bundle-exit PDF,

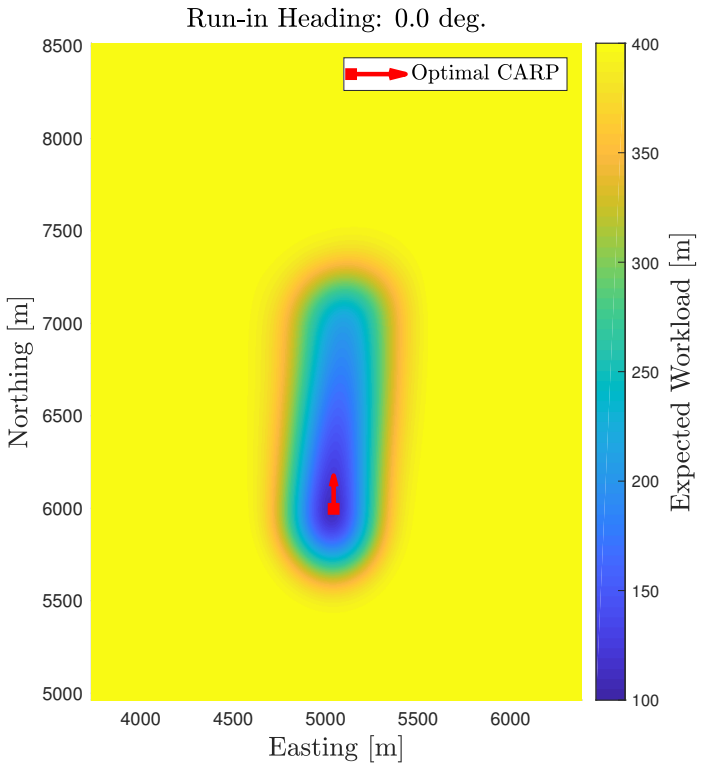
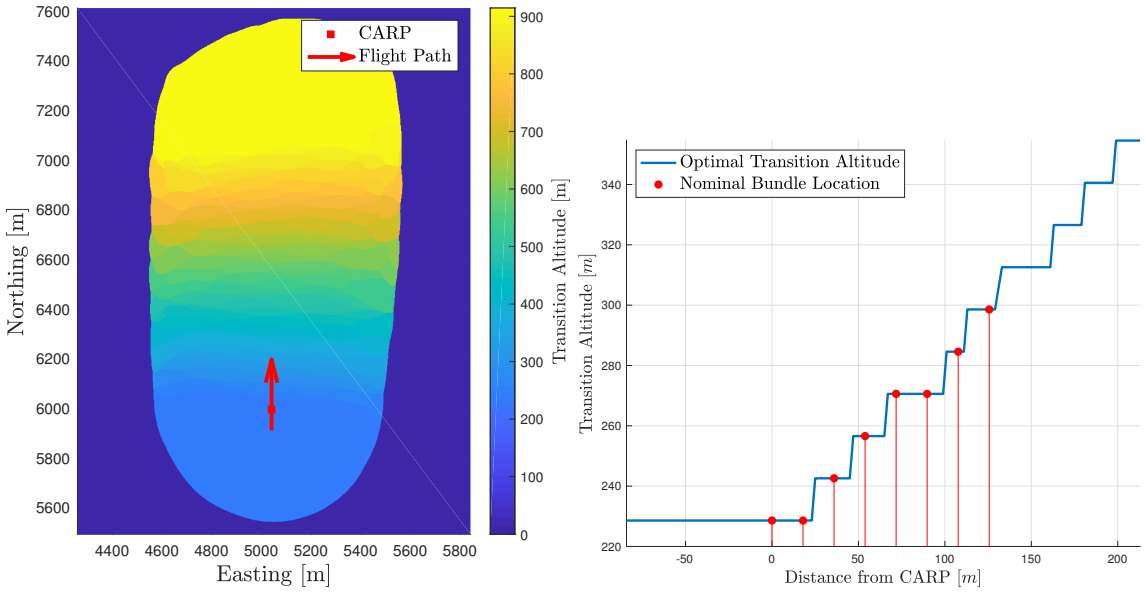


Figure 5.12: Expected Cost function Value (ECV) map with optimal CARP and run-in. Run-in heading is directly into the dominant wind.

conditioned on the CARP and run-in, using the distributions listed in Table 5.2 and the descent parameters drawn from the distributions listed in Table 5.1. The transition altitude of each sample is assigned by evaluating the optimal transition altitude schedule at its xy-location at drop altitude.

Figure 5.14a shows the resulting dispersion of the simulations, overlaid on the Monte Carlo simulation impacts of the predetermined transition scenario (Section 5.1.1). It can be seen that, by allowing the transition altitude to be optimized on a per-bundle case, the resulting dispersion may be decreased. This is most apparent in the region north of the IPI (5000m, 5000m). Given that the dominant wind blows north-to-south, there is not much that can be done to decrease dispersion in the lateral, east-west direction. The total improvement of the optimized transition altitude may be quantified by an empirical CDF, shown in Figure 5.14b. The optimized transition altitude case produces a 50% CEP of 105.1m, an over 30% improvement to the 155.4m CEP produced by the predetermined transition

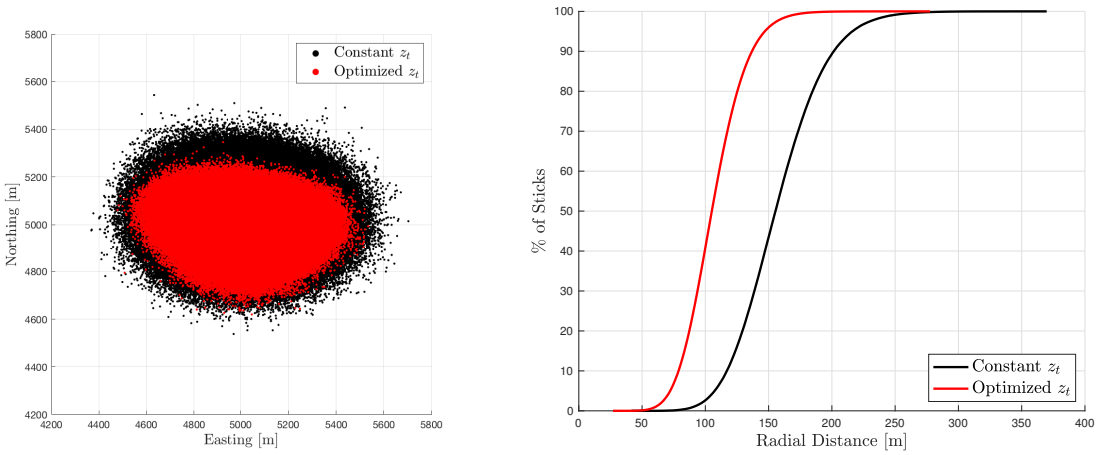


(a) CARP transition altitude schedule

(b) Flightpath transition altitude schedule

Figure 5.13: (left) CARP and flightpath overlaid on transition altitude schedule. (right) The transition altitude schedule evaluated along the flightpath, with the nominal bundle release locations marked.

altitude case. Additionally, the planner predicted an expected workload of 110.6m which, when compared to the Monte Carlo simulation result of 106.9m, represents a relative error of 3.5%.



(a) Monte Carlo xy-impacts

(b) Empirical workload CDF

Figure 5.14: Monte Carlo simulation results for optimized transition altitude of radial workload cost function, IPI = (5000m, 5000m). (left) Impact locations of constant z_t (black) and optimized z_t (red). (right) Empirical CDFs of constant and optimized z_t cases.

5.2.2 Valley Case

In this section, the transition altitude aware CARP and run-in optimization is applied to the complex valley scenario outlined in Section 5.1.2. This scenario, with non-flat terrain and a complex cost function, highlights again the applicability of the developed planning framework to real-world scenarios. Given the non-flat terrain, the transition altitudes are referenced from the $z = 0$ plane. To compensate for the terrain, and to ensure that there is sufficient altitude for the main parachute to fully inflate, the lowest transition altitude is increased to 1300ft above the zero-plane. The maximum transition altitude is kept at 3000ft.

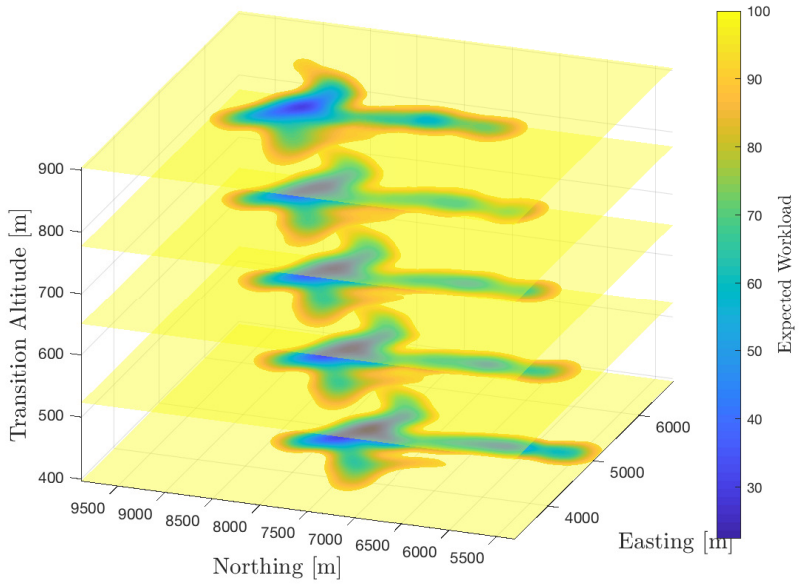
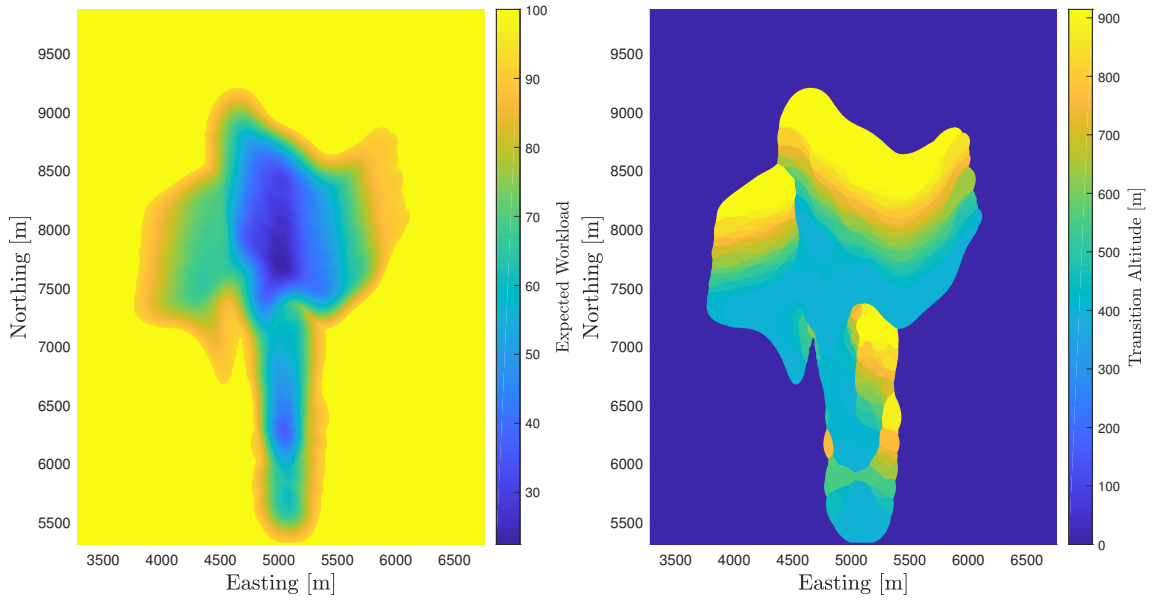


Figure 5.15: Conditional expectation functions for 5 of 50 transition altitudes. Dominant wind is from north to south.

Discretizing the transition altitude range into 50 candidate altitudes, the workload cost function is pulled-back to the drop altitude plane using $350{,}000$ trajectories each (17.5M total). Figure 5.15 shows the conditional expectation functions generated from the sets of trajectories. Like the radial workload case, as the transition altitude increases the minimal workload regions translate against the dominant wind and their expected values increase.



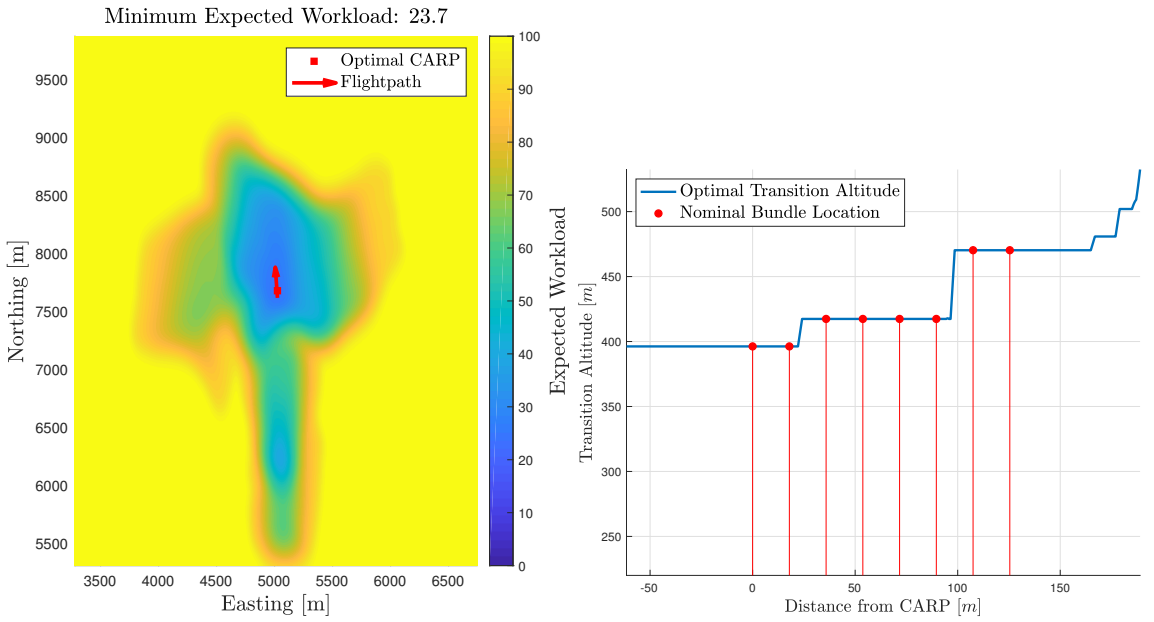
(a) Composite conditional expectation (b) Transition altitude schedule

Figure 5.16: Complex valley scenario with non-flat terrain composite conditional expectation function and corresponding transition altitude schedule *selected* using $\max$ and $\arg\max$, respectively.

The composite conditional expectation function and its corresponding optimal transition altitude schedule are constructed using the $\max$ and $\arg\max$ functions, shown in Figures 5.16a and 5.16b, respectively.

Figure 5.17a shows the ECV map for the optimal run-in heading of -6 deg. The length of the red arrow, the flightpath, is again taken to be the 95% confidence interval to release the 8 bundles. The optimal transition altitude schedule is evaluated along the flightpath and shown in Figure 5.17b. Similar to the radial workload case, the later bundles, i.e. 7 and 8, are scheduled to open earlier (higher) and leverage the dominant wind to be pushed (along their line of control) towards a lower cost area.

To quantify the dispersion, Monte Carlo simulations of $1{,}000{,}000$ 8-bundle sticks (8M simulations in total) were run. Figure 5.18a shows the terrain impacts of the optimized transition altitude case (red) overlaid on the predetermined transition altitude Monte Carlo impacts. While a slight reduction in dispersion can be seen in the northern region of Figure 5.18a, the decrease in the mean workload when including optimized transition altitude



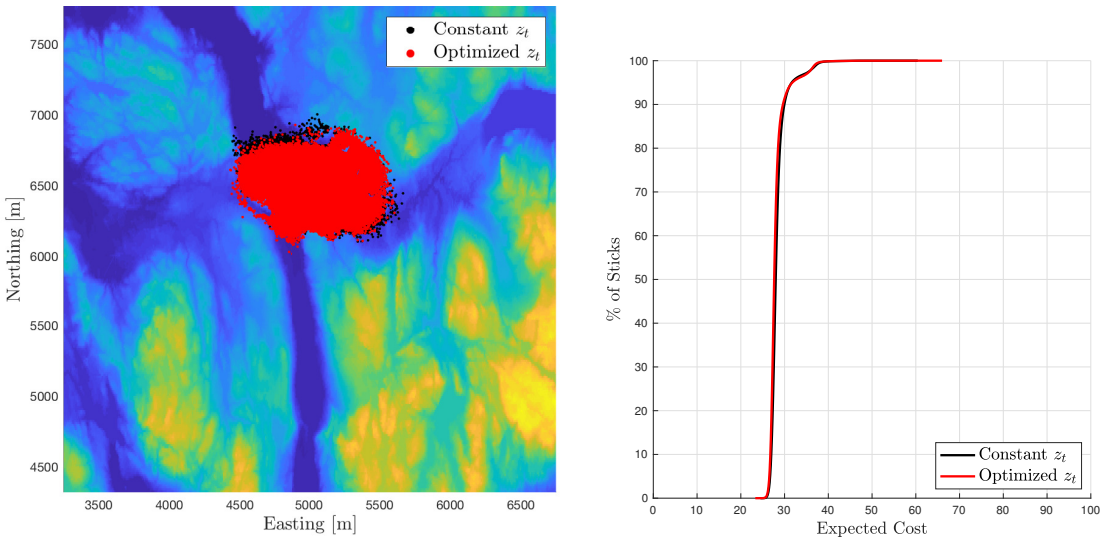
(a) Expected cost function value map

(b) Transition altitude schedule

Figure 5.17: (left) ECV map of optimal run-in heading, -6 deg, with CARP and flightpath overlaid. (right) Optimal transition altitude schedule evaluated along flightpath with nominal-performance bundle release points in red.

is only 0.3 units–down to 28.1 from 28.4. Figure 5.18b shows the empirical CDF of the optimized transition altitude case against that of the predetermined case, the two are nearly identical. Given the relative flatness of the workload cost function in the wide, northern valley, there is not much that may be done to improve the mean workload of the current scenario. However, if the cost function were reconfigured with an IPI in the northern valley, there would be a stronger relationship between the transition altitude and expected cost for the optimization procedure to operate on.

Comparing the planners expected workload, 23.7, to the mean workload from the Monte Carlo simulations, 28.1, shows that the planner underestimated the expected value. The relative error between the two is a sizable 15.7%. However, when using their *scores* (used primarily in intermediate steps) as opposed to cost, the relative error is 6.1%. While there are numerous factors that could contribute to this error, the most probable stems from the steps involving the kernel-based interpolation and integration. As shown in Section 2.2.5, the Lobachevsky spline numerical example, the kernel-based method suffers



(a) Monte Carlo xy-impacts

(b) Empirical workload CDF

Figure 5.18: Monte Carlo simulation results for optimized transition altitude of the complex valley scenario. (left) Impact locations of constant z_t (black) and optimized z_t (red). (right) Empirical CDFs of constant and optimized z_t cases.

from Gibbs phenomenon [81]. Given the sharp increase of the cost function at the valley wall, the interpolated function–and its integrated conditional expectation–would be susceptible to overshoot/undershoot in these regions.

The amount of overshoot/undershoot that the interpolant exhibits can be influenced by the shape parameter α, which can be tuned to help mitigate the phenomenon. However, this is difficult in the optimized transition altitude case as 50 different data-sets, and therefore 50 α values, are considered. The development of an algorithm to intelligently and robustly choose α is left to future research, described in further detail in Section 6.2.2.

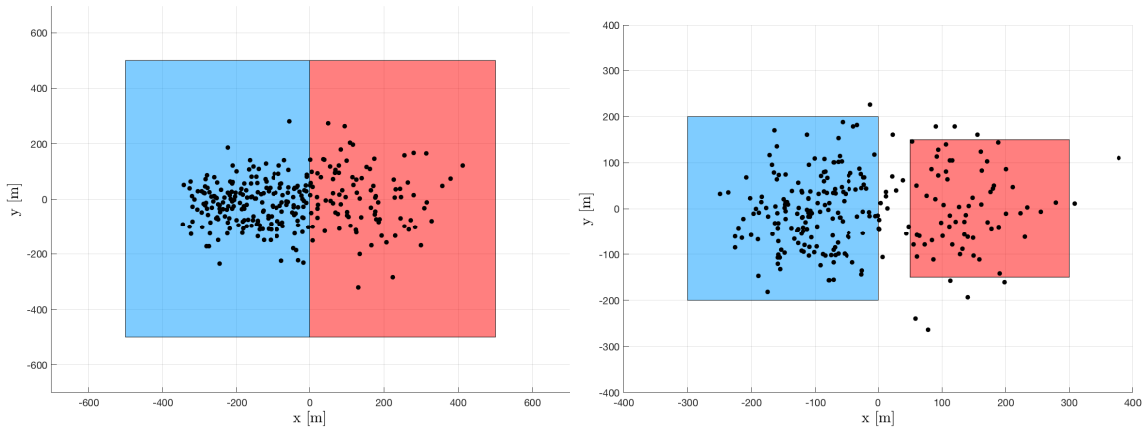
5.3 Solution Set: Real-Time Transition Altitude Optimization

Consider a scenario in which packages are to be dispersed to multiple areas, perhaps several nearby operating bases or several nearby areas for humanitarian aid delivery. In such cases, it is desirable that packages distribute themselves amongst the multiple drop zones according to some pre-specified statistical distribution. To explore this scenario, consider the desired impact PDF to be the union of two disjoint uniform distributions, i.e. $\mathcal{U}_1(x,\, y) \cup \mathcal{U}_2(x,\, y)$, where the magnitude of $\mathcal{U}_1$ is twice that of $\mathcal{U}_2$. Translated into an objective function, the mission objective are represented by

$$g(\boldsymbol{x}) = \begin{cases} \frac{2}{3} & (x, y) \in A_1 \\ \frac{1}{3} & (x, y) \in A_2 \\ 0 & (x, y) \in \mathbb{R}^2 \setminus (A_1 \cup A_2) \end{cases} \tag{5.2}$$

where A_1 and A_2 are the supports of $\mathcal{U}_1$ and $\mathcal{U}_2$, respectively. Using Lebesque integration, the PDF of Eq. (5.2) integrates to 1 and is therefore a valid PDF. However, when integrating over $\mathbb{R}^2$ using the Borel measure, the PDF must be renormalized considering the size (measure) of the supports A_1 and A_2. The PDF, which is treated as an objective function, is pulled-back through a constant wind field of $w_x = 0$m/s and $w_y = 7$m/s to transition altitudes ranging from 240m to 1000m using only the main parachute descent system. The conditional expectation functions are constructed by the methodology presented in [20].

Figure 5.19 shows the results for two different variations in the supports A_1 (blue) and A_2 (red). A Monte Carlo simulations are performed for each, with the drop location taken to be $x_C = 0$m $y_C = -1016$m and random parameters sampled from Table 5.1. The first example considers A_1 and A_2 to be equal in size and adjacent to each other. As such, 2/3 of the packages should be expected to land in A_1 and 1/3 to land in A_2. The Monte Carlo simulations, shown in Figure 5.19a, yielded 202 out of 300 impacts inside A_1 (67.3%) and 98 out of 300 impacts inside A_2 (32.7%). For comparison purposes, a similar Monte Carlo



(a) Scenario 1 (b) Scenario 2

Figure 5.19: Impact dispersion shaping using real-time transition altitude optimization example. Monte Carlo simulation impacts are shown for two examples.

simulation of 300 runs was performed using a constant transition altitude of $z_t = 773$m for all packages. The impact dispersion pattern for this constant z_t case was approximately Gaussian centered at $x = y = 0$, and yielded 158 impacts (52.7%) inside A_1 and 142 impacts (47.3%) inside A_2.

The second example considers two non-adjacent distributions with slightly smaller supports. When re-normalizing Eq. (5.2) for the size of the supports, the resulting impact distribution should yield 75% of impacts within A_1 and 25% within A_2. The Monte Carlo simulations, shown in Figure 5.19b, yielded 166 out of 250 impacts inside A_1 (66.4%) and 59 out of 250 impacts inside A_2 (23.6%), with the rest falling outside of both supports. A similar Monte Carlo simulation with constant $z_t = 773$m yielded 127 impacts (50.8%) inside A_1 , 48 impacts (19.2%) inside A_2, and the remaining 30% outside both supports. Comparing the optimized z_t case with the constant z_t case, the optimized z_t results produced a dispersion pattern much more similar to the desired one. Furthermore, the number of impacts outside of both regions is reduced from 30% in the constant z_t case to 10% in the optimized z_t case. Potentially, the percentage of impacts that missed both supports may be reduced further by increasing N. Overall, the two examples demonstrate the ability of the proposed algorithm to realize desired impact distributions as precisely as possible under a

given degree of uncertainty and computational restrictions.

CHAPTER 6
CONCLUSION

6.1 Contributions

The fundamental work of this thesis is the development of a probabilistic planning methodology with a targeted application of unguided precision airdrop missions. In doing so, a number of novel contributions to the general field of probabilistic, optimal decision making and to the airdrop community have been made. These contributions range from bridging the divide between applied mathematics literature and engineering applications, the development of a generalized decision making framework and its successful application to a real-world problem, and even the creation of unique numerical tools to solve problems otherwise considered intractable.

While the use of the Frobenius-Perron operator has been leveraged in a number of engineering applications [38, 43, 50, 52], the use of the Koopman operator is still limited [33, 34]. Even more notable is that the Frobenius-Perron and Koopman operators' adjoint property has not found use outside of the applied mathematics community. However, for a class of problems considering optimal decision making in the presence of uncertainty, leveraging the Koopman operator through the adjoint property proves to be a valuable tool and a novel contribution of this work.

The generalized optimization procedure developed in Chapter 3.1 shows that when the cost, or objective, function to be optimized is at a future time, the Frobenius-Perron operator may be used to push-forward the initial uncertainty for use in an expected value calculation to probabilistically evaluate the function. Equivalently, the Koopman operator may be used to pull-back the function for use in an expected value calculation at the initial time; where the results are identical. Additionally, using the Method of Characteristics,

the two operators may be applied numerically to nonlinear systems with, possibly, non-Gaussian uncertainty in both states and parameters. However, as detailed in Chapter 3.1, this is where equivalence of the Frobenius-Perron and Koopman approaches begin to diverge. When implemented numerically using the Method of Characteristics, the Frobenius-Perron operator suffers from numerical precision issues given sufficient time [20, 39, 41]. Additionally, when considering a variable other than time as the independent variable, the re-parameterization procedure presented in Section 3.4 is non-trivial for the Frobenius-Perron operator. Through the adjoint relationship leveraged in this work, the Koopman operator approach circumvents both of these issues while returning an equivalent solution, as demonstrated in a spring-mass-damper system simulation example, highlighting again the benefits of this novel approach.

When applied to the unguided precision airdrop problem, the optimization procedure represents a novel planning methodology to determine the optimal CARP, run-in heading, and transition altitude schedule for complex HALO missions. Given the inherent uncertainty in parachute-payload system dynamics and the flight environment, the problem is formulated probabilistically and solved using the Koopman operator approach and the computational benefits it affords. The developed algorithms leverage the adjoint relationship between the Koopman and Frobenius-Perron operators to explicitly compute an expected value characterizing a measure of success, defined by the user, given the uncertainty in the system. The user is returned not only a single optimal answer, but also expected value maps relating all possible CARP and run-in tuples to their expected measure of success for use in decision making, possible re-planning, and overall situational awareness. The planning methodology is differentiated from standard deterministic planners in that the solution is conditioned on the generalized uncertainty in each phase of the mission, and can easily be expanded to new dimensions of uncertainty. In the case of allowing for the transition altitude of the bundles to be optimized alongside the CARP and run-in, the dimension of the control decision vector is carefully increased to ensure that the problem remains tractable.

To assert the feasibility of the developed methodologies through efficient implementation, two numerical toolkits have been developed through this work. The first provides a GPU-accelerated implementation of the kernel-based interpolation and integration methods presented in Section 2.2. In the context of this work, this toolkit is of great benefit for computing conditional expectations and marginals of the high-dimensional, scattered-data. From other work in the field of uncertainty propagation and quantification [52, 51, 41, 39], these problems have been identified but no tractable solutions for high-dimensional problems are presented. The second toolkit provides a GPU-accelerated simulation environment to numerically represent the Frobenius-Perron and Koopman operators through the use of the Method of Characteristics. While demanding *less* trajectories than Monte Carlo methods, given the dimensionality of the unguided precision airdrop problem, the ability to run hundred of thousands to millions of trajectories, in tractable run-time, is needed.

A variety of simulation results demonstrate the practical utility of the planner and provide a quantitative comparison to results generated by a standard deterministic algorithm. Three example scenarios are considered, ranging from a nominal radial-workload case to complex valley and exclude-region cases highlighting th planning methodology's ability to handle complex terrain and constraint elements, respectively. Overall, results show that the proposed probabilistic planner is a promising methodology that can provide operational flexibility and improved outcomes for real-world airdrop missions.

6.2 Recommended Future Research

As is the case with most research, the solving of one problem often leads to new questions to be investigated, or more problems to be solved. The work in this thesis is no different. This Section provides three avenues of research that have arisen during the development of this work. While the included list is not exhaustive, they are considered the open questions that are most interesting to the author and, possibly, the most influential.

6.2.1 Probability-Flux Approach

The definition of the Frobenius-Perron operator enforces the conservation of the probability measure (probability mass) of an element of the state-space, A, as it is mapped by S to $\mathrm{S}(A)$, also an element of $\mathcal{A}$. This measure is induced, using a density, from a known measure, e.g. the Borel measure (the volume of A in the state-space). As the mapping is continually applied, the element travels through the state-space, with its volume contracting or expanding, increasing or decreasing the value of the density on that element as defined by the system dynamics, S. Using the Method of Characteristics, this process is captured along the trajectory of the element.

In the work of Li et al. this is called the *random event description* of the principle of preservation of probability as it is rooted on the trajectory of a random initial condition through the state-space [32]. The authors, however, emphasize another view of phenomenon called the *state space* description. In the state space description, the elements are fixed in their location in the state-space for all time. The element and corresponding boundary are denoted as $A_{\text{fixed}} \in \mathcal{A}$ and $\partial A_{\text{fixed}}$, respectively. The change in probability of A_{fixed} as a function of time is now quantified by the total *flux* through the boundary $\partial A_{\text{fixed}}$.

This description of the problem could prove to be convenient for a number of reasons. First, the set of fixed elements could be chosen to provide a representation of a marginal probability, or conditional expectation, from the outset of the problem. That is, for example, if a problem is defined in $\mathbb{R}^3$ but a conditional expectation across the xy-dimensions is desired, then a set of xy-domains could be defined with their boundaries extending infinitely into the z-dimension creating, in essence, an xy-marginal to be used in the computation of the conditional expectation.

Alternatively, and possibly more interestingly, is defining the elements B in an *observable* space where $\boldsymbol{z} \in B$ such that $\boldsymbol{z} = \beta(\boldsymbol{x}) \ \forall \, \boldsymbol{x} \in A$. For example, given a system whose state vector $\boldsymbol{x}$ contains the velocities $\dot{x}$ and $\dot{y}$, the total velocity may be defined as $V = \beta(\boldsymbol{x}) = \sqrt{\dot{x}^2 + \dot{y}^2}$. Assuming zero initial velocity, the probability that the total ve-

locity is greater than $\lambda > 0$ would be equal to the net flux through the surface defined by $V - \lambda = 0$, or equivalently, the net flux through $\dot{x}^2 - \dot{y}^2 - \lambda^2 = 0$.

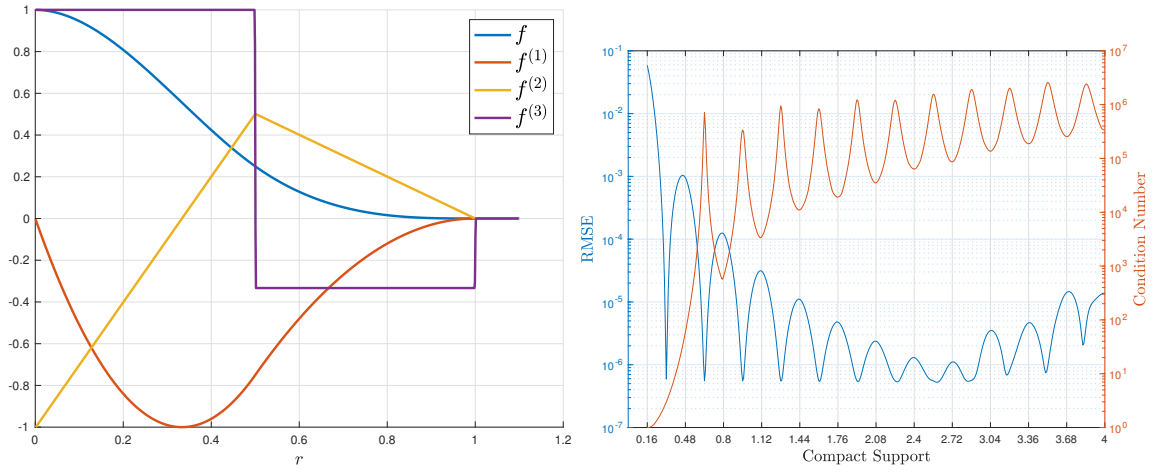
While the state space description (flux), as it is derived in [32], focuses solely on *probabilities*, the use of observables, and corresponding integrals required to enforce conservation, presents an interesting possibility that the approach may be viewed through the Koopman operator.

6.2.2 Robust Selection of Shape Parameter

The use of kernel-based methods for scattered-data interpolation, and more importantly integration, have provided a rigorous, yet convenient, means to computing conditional expectations–an integral component of this work. However, with any data-based method, numerical error is introduced and typically dependent on a characteristic value of the process.

Much of the error associated with the method can be attributed to the choice of α, the *shape* parameter. This parameter, essentially, controls the flatness of the basic function. In this case of compactly support kernels, like the Lobachevsky spline kernel, it controls the size of the compact support. It is shown in the kernel-based approximation literature that, in general, increasing α (leading to a flatter function), decreases the error between the true and interpolated function [62, 61, 71, 72, 73]. However, it is also shown that this increases the condition number, and for compactly supported kernels decreases the sparsity, of the corresponding kernel matrix, $\mathcal{K}$, used in the linear system of equations $\mathcal{K}\boldsymbol{c} = \boldsymbol{g}$ (Section 2.2.2). As such, there exists a *trade-off* between the quality of the solution and the ability to actually compute it [62, 61]. In addition to the shape parameter, the average fill distance (the smallest sphere of radius d about a data-site that does not include another data-site) of all data-sites, is used when quantifying the error in an interpolant.

In its current form, using the compactly supported Lobachevsky spline kernel, this interplay is more complex given their construction using arcs of parabolas of differing degrees,



(a) Lobachevsky L4 (b) RMS error and condition number

Figure 6.1: (left) Lobachevsky L4 ($n = 4$) basic function for $r > 0$ with compact support at $r = 1$. The third derivative is discontinuous at $r = 0.5$. (right) RMSE and condition number when interpolating a 1D Gaussian function using $N = 51$ data-sites against compact support radius.

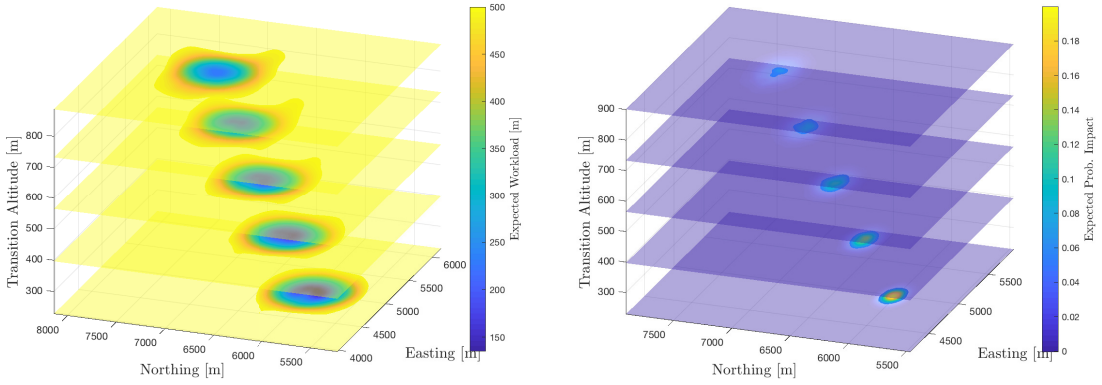
leading to finite smoothness. Figure 6.1a shows the Lobachevsky L4 function ($n = 4$), where discontinuous jumps can be seen in the third derivative at $r = 0.5$ and 1.0. To visualize this relationship, an interpolant was fit using the L4 kernel to a fixed, gridded 1D data-set (constant fill distance between nodes) for a range of shape parameters and the root mean square error (RMSE) and condition number recorded, shown in Figure 6.1b. At α values where the basis function discontinuity occurs in close proximity to a data-site, while the condition number peaks, the RMSE of the interpolant dips. Figure 6.1b shows that even with smaller compact support radii, an approximately constant level of accuracy may be achieved while making the problem computationally easier to solve (smaller condition number).

Based on these 1D trends, data-based heuristics for the *best* selection of α could be developed. Additionally, given the univariate nature of the Lobachevsky kernel, the selection could be applied to each dimension independently. Given that the data is assumed to be scattered, one such algorithm could leverage a frequency plot of the fill distances to compute the α that would maximize the occurrence of placing a discontinuity in close prox-

imity to a data-site. These algorithms would, additionally, help automate the scattered-data interpolation problem by removing the burden of *tuning* α from the user.

6.2.3 Constrained Transition Altitude Optimization

The optimized CARP, run-in heading, and transition altitude methodology has been shown to produce improvements over a predetermined transition altitude for the radial workload scenario and it is expected to improve the constrained, exclude region scenario as well. However, given that the transition altitude control decision is embedded in the system dynamics, whereas the CARP and run-in decisions are found only in the initial condition PDF, the optimal transition altitude selection process must be aware of both the set of objective conditional expectation functions and the set of constraint conditional expectation functions.



(a) Workload cost

(b) Exclude region constraint

Figure 6.2: Pulled-back cost function (left) and constraint function (right) for the constrained exclude region problem.

Previously, in the predetermined transition altitude case, the full expected value calculation for the cost and constraint functions were able to be computed independently and then overlaid, as shown in Figure 5.8 from Section 5.1.3. However, during the optimal transition altitude selection process, for an arbitrary xy-location the candidate transition altitude with the lowest conditional expected cost may also violate the constraint and therefore must be

disregarded. Selecting instead the candidate transition altitude with the lowest expected cost *that also* satisfies the constraint probabilistically.

For example, considering the constrained scenario presented in Section 5.1.3, Figures 6.2a and 6.2b show the sets conditional expectation functions for the pulled-back cost function and constraint function, respectively. Rather than reducing both sets down to single functions and then merging them, they must first be merged–slice for slice– and then reduced. This presents a difficult task. For the unconstrained cases, the optimal transition altitude selection process allowed for an increase in the dimension of the solution space to be mitigated. With the inclusion of the constraint, an algorithm to rigorously, and efficiently, merge the two sets of conditional expectation functions must be developed such that the problem does not become intractable.

Another approach that may be taken is to solve the method of characteristics trajectories using realizations taken at the drop altitude rather than at ground (using a BVP solution). A single set of drop altitude realizations may be used to generate trajectories for each candidate transition altitude along which the objective and constraint functions may be pulled back using the Koopman operator. As such, the optimal transition altitude selection–with knowledge of the constraint–may therefore be applied before the conditional expectation procedure. However, this presents a number of potential problems. First, given that the realizations are taken at the drop altitude, it must be ensured that the corresponding impacts result in a full coverage of the objective space. And second, the selection process may produce an optimal pulled-back objective function that contains jumps, i.e. it is not continuous. Given that this function must be integrated to produce a conditional expectation, this could create difficulties, especially when using the kernel-based methods, e.g. Gibbs phenomenon.

6.2.4 Real-World Airdrop Algorithms

The planning methodology in this work, when applied to the precision airdrop problem, relies on being able to adequately represent the release and descent processes, and their corresponding uncertainty, mathematically. Given the generality of the methodology, improvements in the models or uncertainty distributions may be easily included without much modification. These improvements would help push the planning procedure closer to real-world airdrop missions.

One assumption made in this work is the instantaneous inflation of the drogue parachute as the bundle is released from the aircraft, neglecting the roll-out procedure. Using the developments found in [22], the release process may be included as another stage in the bundles full trajectory to terrain impact. Alternatively, the Frobenius-Perron operator may be used to push forward the bundle exit PDF (Section 4.2.2) through the release dynamics. This propagated quantity could then be used in an expected value equation. Either approach presents an opportunity to capture additional sources of uncertainty in the planning process.

Another opportunity to more closely align the planning procedure with real-world airdrop missions is through the optimal transition altitude selection process (Section 4.3.4). In this work, while system and environmental uncertainty is handled rigorously, it is assumed that the associated densities perfectly represent the uncertainty. In practice, this is not the case, and real-time transition altitude feedback algorithms have been developed to address this [19, 20]. The planning procedure may be made aware of these feedback algorithms by capturing their feedback laws in Υ, the optimal transition altitude selection process. While, in simulation, these selection criteria will be suboptimal when compared to the $\max$, $\arg\max$ pair used in this work, they incorporate the physical limitations of real-world airdrop platforms into the planning process.

6.2.5 Experimental Validation and Data

The control selection methodology developed in this work is formulated such that the results and benefits will be seen in a statistical sense. As such, to experimentally validate the algorithm a large number of trials must be run; much like the simulations performed during Monte Carlo validation. This is typical for stochastic optimization problems and is often difficult to accomplish. For the methodology's application to precision airdrop, this is largely impractical due to constraints such as time, costs, and repeatability. However, on a simpler systems, such as the spring-mass-damper system presented in Section 3.5, it could be feasible to experimentally validate the methodology itself and against other solutions, e.g. deterministic.

The collection of experimental data does present another possible avenue of research in the form investigating the use of dynamic mode decomposition [33, 34] as a part of the optimization methodology. In this work, the Koopman operator of a system is represented by evaluating its method of characteristics solution along a number of simulated trajectories. Alternatively, dynamic mode decomposition (DMD) is a data-driven approach that seeks to represent the Koopman operator numerically by leveraging its linearity property. The linearity property states that while $\mathrm{U_S}$, from $\mathrm{U_S}g(\boldsymbol{x}) = g(\mathrm{S}(\boldsymbol{x}))$, may be infinite dimensional it is always linear [31]. By truncating to its dominant modes, the operator may be represented by a matrix using DMD.

While DMD is derived using the Koopman operator (Koopman theory), it does not use, nor make mention of, its adjoint relationship to the Frobenius-Perron operator. It would therefore be an interesting pursuit to develop a methodology leveraging both, by substituting a Koopman operator developed from experimental data using DMD in place of the simulated MOC trajectories for use in the control selection under uncertainty problem. This would provide a means to apply the methodology developed in this work to complex, real-world systems where accurate modeling is difficult but experimental data is available or able to be collected.

REFERENCES

[1] P. Kumar, *Stochastic systems : estimation, identification, and adaptive control*. Englewood Cliffs, NJ: Prentice-Hall Inc., 2016.

[2] G. Fabbri and F Gozzi, *Stochastic Optimal Control in Infinite Dimension: Dynamic Programming and HJB Equations*. Cham: Springer International Publishing, 2017, pp. 91–170.

[3] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Baisc Engineering*, vol. 82, pp. 35–45, 1960.

[4] A. Gelb, *Applied Optimal Estimation*. MIT Press, 1974.

[5] J. S. Gibson, *Numerical approximation for the infinite-dimensional discrete-time optimal linear-quadratic regulator problem*, ser. NASA contractor report ; NASA CR-178081. 1986.

[6] B. D. O. Anderson, *Optimal control : linear quadratic methods*, ser. Prentice Hall information and system sciences series. Englewood Cliffs, N.J.: Prentice Hall, 1990.

[7] G. Williams, A. Aldrich, and E. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344 –357, 2017.

[8] J. Westmann and F. Hanson, "Computational method for nonlinear stochastic optimal control," in *American Control Conference, 1999. Proceedings of the 1999*, 1999.

[9] J. Rogers and N. Slegers, "Robust parafoil terminal guidance using massively parallel processing," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 5, pp. 1336–1345, 2013.

[10] J. Rogers, "Massively-parallel stochastic control and automation: A new paradigm in robotics," in *GPU Technology Conference*, San Jose, CA, Mar. 2014.

[11] E. Todorov and W. Li, "A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proceedings of the 2005, American Control Conference, 2005.*, 2005, pp. 300–306.

[12] X. Luan, F. Liu, and P. Shi, "Neural network based stochastic optimal control for nonlinear markov jump systems," *International Journal of Innovative Computing, Information and Controls*, vol. 6, no. 2, pp. 3715–3723, 2010.

[13] A. Prabhakar, J. Fisher, and R. Bhattacharya, "Polynomial chaos-based analysis of probabilistic uncertainty in hypersonic flight dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 1, pp. 222 –234, 2010.

[14] R. Mcgraw, "Description of aerosol dynamics by the quadrature method of moments," *Aerosol Science and Technology*, vol. 27, no. 2, pp. 255 –265, 1997.

[15] M. Kumar, P. Singla, S. Chakravorty, and J. Junkins, "A multi-resolution approach for steady state uncertainty determination in nonlinear dynamical systems," in *Proceedings of the Annual Southeastern Symposium on System Theory*, 2006, pp. 344–348.

[16] D. Simon, *Optimal State Estimation:Kalman, H Infinity, and Nonlinear Approaches*. Hoboken, NJ: John Wiley & Sons, 2006.

[17] J. Sadeck, J. Riley, K. Desabrais, and C. Lee, "Low-cost halo cargo airdrop systems," in. American Institute of Aeronautics and Astronautics, 2009.

[18] R. Benney, J. Barber, J. McGrath, J. McHugh, G. Noetscher, and S. Tavan, "The new military applications of precision airdrop systems," in. American Institute of Aeronautics and Astronautics, 2005.

[19] A. R. Gerlach and D. B. Doman, "Analytical solution for optimal drogue-to-main parachute transition altitude for precision ballistic airdrops," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 439–452, 2016.

[20] A. Leonard, B. Klein, C. Jumonville, J. Rogers, A. R. Gerlach, and D. B. Doman, "A Probabilistic Algorithm for Ballistic Parachute Transition Altitude Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 12, pp. 3037–3049, 2017.

[21] P. Welsh, "Team works to improve precision of high altitude airdrops," in. 66th Air Base Group Public Affairs, 2013.

[22] J. T. VanderMey, D. B. Doman, and A. R. Gerlach, "Release Point Determination and Dispersion Reduction for Ballistic Airdrops," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 11, pp. 2227–2235, 2015.

[23] J. Potvin, R. Charles, and K. Desabrais, "Comparative DSSA Study of Payload-Container Dynamics Prior to, During and After Parachute Inflation," in *19th AIAA*

Aerodynamic Decelerator Systems Technology Conference and Seminar, AIAA, May 2007.

[24] P. Cuthbert, "A Software Simulation of Cargo Drop Tests," in *17th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, AIAA, May 2003.

[25] M. R. Cacan, E. Scheuermann, M. Ward, M. Costello, and N. Slegers, "Autonomous airdrop systems employing ground wind measurements for improved landing accuracy," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 6, pp. 3060–3070, 2015.

[26] C. Munnell, "Company developing wind measurement technology to improve cargo airdrops," *National Defense Magazine*, 2014.

[27] R. Wright, R. Benney, and J. McHugh, "An on-board 4d atmospheric modeling system to support precision airdrop," in. American Institute of Aeronautics and Astronautics, 2005.

[28] D. Cambell, T. Fill, P. Hattis, and S. Tavan, "An on-board mission planning system to facilitate precision airdrop," in. American Institute of Aeronautics and Astronautics, 2005.

[29] N. Slegers, J. D. Rogers, and A. Brown, "Experimental investigation of stochastic parafoil guidance using a graphics processing unit," in. American Institute of Aeronautics and Astronautics, 2015.

[30] A. Lasota, *Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics*, 2nd ed.., ser. Applied mathematical sciences (Springer-Verlag New York Inc.) ; v. 97. New York: Springer-Verlag, 1994.

[31] B. O. Koopman, "Hamiltonian systems and transformation in hilbert space," *Proceedings of the National Academy of Sciences*, vol. 17, no. 5, pp. 315–318, 1931.

[32] J. Li and J. Chen, *Stochastic Dynamics of Structures*, 2nd ed.. New York: John Wiley & Sons, 2009, ISBN: 9780470824245.

[33] J. Kutz, S. Brunton, B. Brunton, and J. Proctor, *Dynamic Mode Decomposition*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2016.

[34] J. Proctor, S. Brunton, and J. Kutz, "Dynamic mode decomposition with control," *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 1, pp. 142–161, 2016.

[35] A. R. Gerlach, S. G. Manyam, and D. B. Doman, "Precision airdrop transition altitude optimization via the one-in-a-set traveling salesman problem," in *2016 American Control Conference (ACC)*, 2016, pp. 3498–3502.

[36] *Air force future operating concept*, United States Air Force, 2015.

[37] R. S. Park and D. J. Scheeres, "Nonlinear mapping of gaussian statistics: Theory and applications to spacecraft trajectory design," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1367–1375, 2006.

[38] A. Halder, K. Lee, and R. Bhattacharya, "Optimal transport approach for probabilistic robustness analysis of f-16 controllers," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 10, pp. 1935–1946, 2015.

[39] A. Probe, T. A. Elgohary, and J. L. Junkins, "A new method for space objects probability of collision," in *AIAA/AAS Astrodynamics Specialist Conference, AIAA SPACE Forum*, AIAA, Long Beach, California, Sep. 2016.

[40] P. Dutta, A. Halder, and R. Bhattacharya, "Uncertainty quantification for stochastic nonlinear systems using perron-frobenius operator and karhunen-loève expansion," in *Control Applications (CCA), 2012 IEEE International Conference on*, 2012, pp. 1449–1454.

[41] R. Hoogendoorn, E. Mooij, and J. Geul, "Uncertainty propagation for statistical impact prediction of space debris," *Advances in Space Research*, vol. 61, no. 1, pp. 167 –181, 2018.

[42] T. T. Soong, *Random Differential Equations in Science and Engineering*, ser. Mathematics in Science and Engineering. New York: Academic Press, 1973.

[43] A. Y. Weiße, "Global sensitivity analysis of ordinary differential equations," PhD thesis, Freien Universität Berlin, 2009.

[44] D. L. Cohn, *Measure theory*. Boston: Birkhäuser, 1980.

[45] P. G. Hoel, *Introduction to probability theory*, ser. The Houghton-Mifflin series in statistics. Boston: Houghton Mifflin, 1971.

[46] K. L. Chung, *A Course in Probability Theory*, 3rd ed. San Diego: Academic Press, 2001, ISBN: 0121741516.

[47] B. Epstein, *Partial Differential Equations*. New York: McGraw-Hill, 1962.

[48] K. Sobczyk, *Stochastic differential equations : with applications to physics and engineering*, ser. Mathematics and its applications. Dordrecht: Kluwer Academic, 1991.

[49] H. L. Vasudeva, *Elements of Hilbert spaces and operator theory*. 2017.

[50] P. Dutta, A. Halder, and R. Bhattacharya, "Nonlinear estimation with perron-frobenius operator and karhunen-love expansion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 4, pp. 3210–3225, 2015.

[51] A. Halder, "Probabilistic Methods for Model Validation," PhD thesis, Texas A&M University, 2014.

[52] A. Halder and R. Bhattacharya, "Dispersion analysis in hypersonic flight during planetary entry using stochastic liouville equation," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 459–474, 2011.

[53] A. Y. Weiße, R. H. Middleton, and W. Huisinga, "Quantifying uncertainty, variability and likelihood for ordinary differential equation models," *BMC Systems Biology*, vol. 4, no. 1, p. 144, 2010.

[54] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[55] B Chazelle, "An optimal convex-hull algorithm in any fixed dimension," *Discrete & Computational Geometry*, vol. 10, no. 4, pp. 377–409, 1993.

[56] B. Scholkopf, K.-K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with gaussian kernels to radial basis function classifiers," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2758–2765, 1997.

[57] S. S. Haykin, *Neural networks and learning machines*, 3rd ed.. New York: Prentice Hall-Pearson, 2009.

[58] A. Masood, A. M. Siddiqui, and M. Saleem, "A radial basis function for registration of local features in images," in *Advances in Image and Video Technology*, D. Mery and L. Rueda, Eds., Berlin, Heidelberg: Springer, 2007, pp. 651–663.

[59] G. Allasia, R. Cavoretto, and A. De Rossi, "Local interpolation schemes for landmark-based image registration: A comparison," *Mathematics and Computers in Simulation*, vol. 106, pp. 1 –25, 2014.

[60] A. R. Gerlach, "Autonomous path-following by approximate inverse dynamics and vector field prediction," PhD thesis, University of Cincinnati, 2014.

[61] G. E. Fasshauer and M. McCourt, *Kernel-based Approximation Methods using MATLAB*, ser. Interdisciplinary Mathematical Sciences. World Scientific, 2016.

[62] G. E. Fasshauer, *Meshfree Approximation Methods with MATLAB*, ser. Interdisciplinary Mathematical Sciences. World Scientific, 2007.

[63] H. Wendland, *Scattered Data Approximation*. Cambridge, UK: Cambridge University Press, 2005.

[64] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed., ser. Johns Hopkins studies in the mathematical sciences. Baltimore: Johns Hopkins University Press, 2013.

[65] C. T Kelley, *Iterative Methods for Linear and Nonlinear Equations*. Philadelphia: Society for Industrial and Applied Mathematics, 1995.

[66] M. D. Buhmann, *Radial Basis Functions Theory and Implementations*, ser. Cambridge monographs on applied and computational mathematics ; 12. Cambridge ; New York: Cambridge University Press, 2003, ISBN: 1-280-43229-2.

[67] W. H. Press, *Numerical recipes in FORTRAN 77 : the art of scientific computing*, 2nd ed. Cambridge, UK: Cambridge University Press, 1992.

[68] M. J. D. Powell, "Radial basis functions for multivariate interpolation: A review," ser. Algorithms for approximation, Oxford University Press, 1987.

[69] A. Sommariva and M. Vianello, "Numerical cubature on scattered data by radial basis functions," *Computing*, vol. 76, no. 3, 2005.

[70] R. Cavoretto, "Meshfree approximation methods, algorithms and applications," PhD thesis, University of Turin, 2010.

[71] G. Allasia, R. Cavoretto, and A. De Rossi, "Lobachevsky spline functions and interpolation to scattered data," *Computational and Applied Mathematics*, vol. 32, no. 1, pp. 71–87, 2013.

[72] ——, "Numerical integration on multivariate scattered data by lobachevsky splines," *International Journal of Computer Mathematics*, vol. 90, no. 9, pp. 2003 –2018, 2013.

[73] ——, "Multidimensional lobachevsky spline integration on scattered data," *Applied Mathematics & Information Sciences*, vol. 8, no. 1, pp. 145 –151, 2014.

[74] G. Allasia, R. Cavoretto, A. De Rossi, B. Quatember, W. Recheis, M. Mayr, and S. Demertzis, "Radial basis functions and splines for landmark based registration of medical images," *AIP Conference Proceedings*, vol. 1281, no. 1, pp. 716–719, 2010.

[75] A. Leonard, J. Rogers, A. R. Gerlach, and D. B. Doman, "A probabilistic approach to the optimal determination of a computed air release point," in *24th AIAA Aerodynamic Decelerator Systems Technology Conference*, Denver: AIAA, Jun. 2017.

[76] A. Leonard, J. Rogers, and A. R. Gerlach, "Koopman operator approach to airdrop mission planning under uncertainty," *Manuscript submitted for publication*, 2019.

[77] J. E. Stone, D. Gohara, and G. Shi, "Opencl: A parallel programming standard for heterogeneous computing systems," *Computing in Science Engineering*, vol. 12, no. 3, pp. 66–73, 2010.

[78] I. Pohl, *Object-Oriented Programming using C++*, 2nd ed.., ser. The Addison-Wesley series in Object-Oriented Software Engineering. Addison Wesley, 1997.

[79] R. Franke, "A critical comparison of some methods for interpolation of scattered data," Naval Postgraduate School, Tech. Rep. NPS-53-79-003, Mar. 1979.

[80] G. Leobacher and F. Pillichshammer, *Introduction to Quasi-Monte Carlo Integration and Applications*, ser. Compact Textbooks in Mathematics. Springer International Publishing, 2014.

[81] B. Fornberg, T. Driscoll, G. Wright, and R. Charles, "Observations on the behavior of radial basis function approximations near boundaries," *Computers & Mathematics with Applications*, vol. 43, no. 3, pp. 473 –490, 2002.

[82] J. Rogers, A. Leonard, C. Jumonville, A. Gerlach, and D. Doman, "Shaping dispersion patterns in complex dropzones through parachute transition altitude optimization," in *24th AIAA Aerodynamic Decelerator Systems Technology Conference*, AIAA, Denver, Jun. 2017.

[83] C Barber, D. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[84] H. Edelsbrunner and E. Mcke, "Three-dimensional alpha shapes," *ACM Transactions on Graphics (TOG)*, vol. 13, no. 1, pp. 43–72, 1994.

[85] W. Ludtke, "A technique for the calculation of the opening-shock forces for several types of solid cloth parachutes," in *4th AIAA Aerodynamic Decelerator Systems Technology Conference*, Palm Springs, CA: AIAA, May 1973.

[86] T. D. Fields, J. C. LaCombe, and E. L. Wang, "Autonomous guidane of a circular parachute using descent rate control," *Journal of Guidance, Control, and Dynamics*, Jul. 2012.

[87] A. Cinnamon, "Improving airdrop precision through error budget analysis," in *Interservice/Industry Training, Simulation, and Education Conference*, NTSA, Orlando, 2016.

[88] G. A. Petry, "Airdrop error analysis," Wright Patterson Air Force Base, Ohio, Air Force Systems Command Technical Report ASD-TR-75-8, Jun. 1975.

[89] A. R. Gerlach, D. B. Doman, J. Rogers, and A. Leonard, "Probabilistic airdrop planning for dynamic drop zones," in *24th AIAA Aerodynamic Decelerator Systems Technology Conference*, AIAA, Denver, Jun. 2017.

[90] T. H. Cormen, *Introduction to algorithms*, 3rd ed. Cambridge, Mass.: MIT Press, 2009, ISBN: 9780262270830.

[91] J. A. Sethian, *Level set methods and fast marching methods : evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Cambridge: Cambridge, 1999.

[92] R. M. M. Mattheij, R. D. Russell, and U. M. Ascher, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Philadelphia, Pa.: Society for Industrial and Applied Mathematics, 1995.

[93] E. Hairer, *Solving ordinary differential equations II*, 2nd ed.., ser. Springer series in computational mathematics. Springer, 2010.

[94] J. C. Butcher, *Numerical methods for ordinary differential equations*, 3rd ed.. 2016.

[95] D. Demidov, K. Ahnert, K. Rupp, and P. Gottschling, "Programming CUDA and OpenCL: A case study using modern C++ libraries," *SIAM Journal on Scientific Computing*, vol. 35, no. 5, pp. 453–472, 2013.

[96] K. Ahnert, D. Demidov, and M. Mulansky, "Solving ordinary differential equations on GPUs," in *Numerical Computations with GPUs*, Springer International Publishing, 2014, pp. 125–157, ISBN: 9783319065489.

[97] K. Ahnert and M. Mulansky, "Odeint - solving ordinary differential equations in C++," *AIP Conference Proceedings*, vol. 1389, no. 1, pp. 1586–1589, 2011.

[98] L. F. Shampine and M. W. Reichelt, "The MATLAB ODE suite," *SIAM Journal on Scientific Computing*, vol. 18, no. 5, pp. 1–22, 1997.

[99] National Imagery and Mapping Agency (NIMA), "Performance specifications: Digital Terrain Elevation Data (DTED)," Washington, DC, Tech. Rep. MIL-PRF-89020B, 2000.

[100] W. Skamarock, J. Klemp, J. Dudhia, and D. Gill, "A Description of the Advanced Research WRF Version 3," Boulder, CO, NCAR Technical Note TN-475+STR, Jun. 2008.

[101] B. Klein and J. D. Rogers, "A Probabilistic Approach to Unguided Airdrop," in *23rd AIAA Aerodynamic Decelerator Systems Technology Conference*, AIAA, Apr. 2015.

VITA

Andrew Leonard was born and raised in West Grove, PA. In May 2014 he received a Bachelor of Science in Mechanical Engineering from the Pennsylvania State University. In August of the same year, he entered the Woodruff School of Mechanical Engineering at the Georgia Institute of Technology as part of Joint M.S program with Universität Stuttgart in Stuttgart, Germany. From May to August 2015 Andrew participated in a Summer Research program at the Air Force Research Lab at Wright-Patterson Air Force Base. Immediately following his time at AFRL, Andrew traveled to Stuttgart to continue the second half of his M.S program, completing his Master's thesis through an internship with Daimler AG. Andrew returned to Georgia Tech in September 2016 to pursue his PhD. His technical interests include planning under uncertainty, dynamic modeling and controls, and the application of GPU-computing to engineering problems.